Journal of Computing and Electronic Information Management
ISSN: 2413-1660 | Vol. 23, No. 2, 2026

Dynamic Resource Scheduling for Real-Time Multimodal Inference on Heterogeneous Edge Devices

-- A Trace-Calibrated Study of Deadline-Aware Audio–Visual Scheduling

Aisling O'Connor*

School of Computer Science, University College Dublin, Dublin, Ireland

*** Corresponding author**

Abstract: Real-time multimodal inference couples resource-intensive visual and acoustic feature extraction with latency-sensitive fusion. Heterogeneous edge nodes can offer unequal execution times, while static routing ignores queue states and request deadlines. This paper investigates a low-overhead dynamic routing scheme, EDF-ECT, which combines estimated completion-time placement with non-preemptive earliest-deadline-first local queues. A compact dual-branch convolutional graph is timed on a CPU using three threading configurations; 432 measured invocations calibrate three virtual service profiles. An event-driven replay evaluates six policies using workload-adjusted arrival rates and common per-request service samples. Across eight independent traces per setting at nominal load 0.95, EDF-ECT improves the deadline-hit rate from 94.57% to 97.07% for balanced traffic and from 96.92% to 99.27% for large-input-heavy traffic. Paired 95% confidence intervals for the gains are [1.73, 3.26] and [1.96, 2.75] percentage points. Deadline gains trade off against unconditional p95 response time. The findings provide a reproducible request-level scheduler study with measured service kernels while distinguishing virtual workers from physical edge devices.

Keywords: Edge intelligence; Multimodal inference; Dynamic scheduling; Earliest deadline first; Heterogeneous computing; Performance simulation.

1. Introduction

Edge-side inference shortens the distance between sensor inputs and prediction services, but computational budgets vary among deployment nodes. Foundational edge and cloudlet studies motivate localized execution, and surveys highlight the latency, energy, and network constraints that distinguish this setting from centrally provisioned clouds [1]–[4]. Collaborative inference frameworks subsequently developed partitioning, offloading, and adaptation mechanisms [5]–[12]. Those systems illustrate why a per-request scheduling decision must account for both input demand and available execution capacity.

Multimodal applications further complicate the scheduling problem because a single request may contain an image and an acoustic segment, and both branches must complete before fusion is available. Higher image resolution and audio-segment length affect service times differently. At an edge gateway, incoming jobs can carry distinct deadline budgets even when using the same inference architecture. A policy that merely sends requests to the fastest measured node may overload it; a policy that balances queue lengths may allocate a computationally heavy job to a slow node. Therefore, completion-time-aware routing and deadline-aware execution order address two related but separate aspects of real-time performance.

This paper presents EDF-ECT, a lightweight policy that selects a virtual node using predicted end-to-end completion time and reorders its waiting jobs by absolute deadline. The system is evaluated through hardware-measured inference kernels and event-driven replay, rather than through claims of deployment on distinct edge boards. Its contribution is a compact scheduling implementation with four conventional routing baselines, a bounded-EDF ablation, and evidence about the conditions in which deadline prioritization exchanges response-time tail behavior for higher service-level compliance.

The scope is deliberately request-level resource scheduling. Model training, inference accuracy, networking, batching and processor power regulation are outside the evaluated service boundary. This separation provides a targeted characterization of queue management that can be calibrated later with device-specific profiles without changing the dispatcher.

2. Background and related work

2.1. Edge and Efficient Inference

Embedded inference research includes model decomposition, early exits, adaptive evaluation and compact neural architectures [7], [13]– [19]. DeepX dynamically allocates execution to mobile processors [7]; BranchyNet and adaptive inference shorten selected prediction paths [13], [14]; compression and mobile model families reduce resource demand [15]– [19]. In contrast, the present dispatcher treats the model graph and its input-dependent service profile as fixed. It changes which waiting request executes next and which virtual node receives new work, leaving model quality unchanged by scheduling.

2.2. Multimodal Workloads

Multimodal learning encompasses representation, alignment and fusion [20]. Transformer-based and contrastive systems demonstrate image–language, video–audio–text, and image–audio associations [21]–[26]. Their architectures motivate a dual-modality scheduling abstraction in which feature extraction from the two inputs precedes fusion. The evaluation uses a small audio–visual convolutional graph to

isolate compute and queueing effects; it does not claim the recognition accuracy of those pretrained systems.

2.3. Cluster and Serving Schedulers

Large-scale learning schedulers address resource placement, fairness and job execution efficiency [27]–[29]. Serving platforms such as Clipper, Clockwork, INFaaS, Nexus and InferLine employ batching, profiling, placement or proactive capacity management to meet latency targets [30]–[34]. Work on video pipelines adapts model configurations and stream resource allocations [35]–[37]; Ray demonstrates general-purpose dynamic task execution [38]. Our narrow setting considers non-preemptive inference jobs on three independently modeled serial workers. It does not reproduce any of the larger systems; it compares simple, fully specified routing and local queue rules under a common measured service trace.

3. System and workload model

3.1. Audio–Visual Inference Graph

Each request j contains one three-channel image and one one-channel waveform. A three-layer 2-D convolutional image encoder and a three-layer 1-D convolutional audio encoder produce pooled features of 48 and 32 dimensions, respectively. Concatenation produces an 80-dimensional vector processed by two linear layers. The graph is in inference mode, with deterministic randomly initialized parameters; training labels, pretrained weights and predictive accuracy are not used because the dependent variables are execution time and deadline completion. The small, medium and large classes use image sizes 96×96, 144×144 and 192×192 with audio lengths 4096, 8192 and 12288 samples, respectively.

Jobs are non-preemptive after they start, and the full dual-branch inference is modeled as one serial service event. This avoids attributing unavailable measurements to branch-wise parallelism. A calibrated service time is selected from actual end-to-end kernel measurements for the assigned class and virtual-node profile.

3.2. Virtual Heterogeneous Nodes

The executable measurement environment exposes five logical Intel Xeon Platinum 8573C CPUs and PyTorch 2.10.0+cpu. The same host runs the identical inference graph at 1, 2 and 4 intra-operation CPU threads, establishing profiles d=1, 2 and 3. Each cell contains eight warm-up invocations and 48 timed invocations measured by a monotonic high-resolution clock. Table I reports arithmetic means and empirical 95th percentiles. Threading does not always accelerate small kernels, so the observed profiles are intentionally treated as measured non-monotonic capacities, not ranked hardware speed classes.

Table I. Measured kernel latencies (mean / p95, ms)

Threads	Small	Medium	Large
1	1.20/2.43	1.35/2.30	1.74/2.87
2	3.76/8.10	15.80/38.07	4.87/10.07
4	11.08/24.12	18.64/67.41	3.29/8.58

In the replay, these three profiles behave as independent virtual nodes with one running job each. They are not three physically networked edge devices; communication time, hardware contention across nodes, bandwidth and power draw are not modeled. The measured sample arrays are nevertheless used directly for service-time draws, rather than replacing them with invented per-device times.

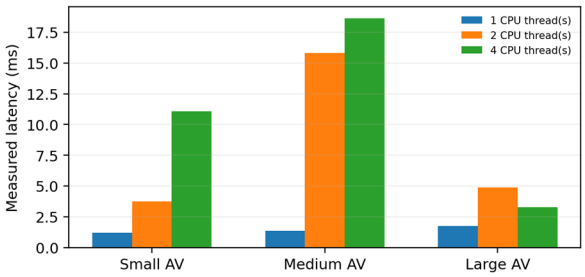


Fig. 1 Mean measured kernel latency for three input classes and CPU-thread profiles.

3.3. Arrival Process and Deadlines

We use three request classes with independent exponential interarrival times. Balanced traffic chooses each class with probability 1/3; large-heavy traffic uses probabilities (0.15, 0.25, 0.60). The absolute deadline D_j equals arrival time a_j plus 6, 12 or 22 ms for small, medium or large requests. These are explicit synthetic service objectives, not externally asserted application requirements. The offered-load parameter ρ takes values 0.55, 0.75, 0.95 and 1.15. For each traffic mix with class probabilities p_c, let m_d=∑_c p_c μ_{d,c} and K=∑_d 1000/m_d requests/s. The arrival rate is λ=ρK and ρ∈{0.55, 0.75, 0.95, 1.15}. Thus the offered-load labels use the respective class mix rather than a capacity estimated under another mix; K is a nominal aggregate service reference, not actual achieved throughput under a scheduler.

Each case uses 2800 generated requests and eight fixed random seeds. The first 10% of arrivals are excluded from reported metrics to reduce initialization effects. Before policy evaluation, a per-request/per-worker service duration is drawn with replacement from the 48 observed samples for that class and worker. This fixed matrix and the common arrival trace are reused across all six policies for a matched comparison. The primary metric is deadline satisfaction, H = (1/N)∑_j 𝟙[C_j ≤ D_j], with C_j the completion timestamp. Mean response time, 95th/99th-percentile response time, waiting queue length and throughput are secondary descriptors.

4. Dynamic scheduling method

4.1. Estimated Completion-Time Placement

Let μ_{d,c} denote the mean measured execution time for class c on node d. At arrival a_j, r_d is the remaining service time of the running request on node d, and Q_d is its waiting set. Under FIFO, a prospective completion estimate is F_FIFO(j,d) = a_j + r_d + W_FIFO(d) + μ(d,c_j), where W_FIFO(d) sums the estimated service times of the waiting jobs and μ(d,c_j) is the incoming request mean measured service time. The ECT-FIFO baseline routes j to the node minimizing this expression and appends j to the queue. This is a queue-aware approximation to minimum finish-time assignment, related to classic heterogeneous scheduling ideas [40].

4.2. EDF-ECT Routing and Local Ordering

The proposed policy assigns each queued request an absolute deadline and, whenever a worker becomes idle,

selects the waiting request having the earliest deadline. Since executing jobs cannot be interrupted, the running job remains fixed. To estimate the completion of incoming job j under EDF, only queued jobs with deadlines no later than D_j are assumed to precede it. The placement estimate is F_EDF(j,d) = a_j + r_d + W_EDF(j,d) + μ(d,c_j), where W_EDF(j,d) sums estimated service times only for queued requests with D_k ≤ D_j. The dispatcher chooses the node minimizing F_EDF(j,d), inserts the new job, and applies non-preemptive EDF ordering at the next worker release.

EDF is a standard deadline-driven scheduling principle [39], while the earliest-finish placement component uses heterogeneous measured costs [40]. Their combination here is intentionally compact: the implementation stores each queue as a list and scans all three nodes at admission. For D devices and at most Q waiting jobs per device, a straightforward dispatch costs O(DQ), with O(Q log Q) local sorting only when selecting an EDF job. No online model fitting, migration or task splitting is required.

4.3. Baselines and Bounded-EDF Ablation

RoundRobin cycles through workers independently of service profiles. Fastest selects the lowest mean-time node for the arriving class, ignoring its queue. JSQ chooses the fewest outstanding jobs. ECT-FIFO minimizes the FIFO predicted finish. EDF-ECT uses the EDF-specific predicted finish and deadline ordering. An additional BA-EDF ablation applies EDF only when the local waiting list has at most two requests and otherwise restores arrival order. All six policies share identical request timestamps, classes, and pre-sampled per-request/per-worker execution times within each seed. Deadline ties are resolved by request index; routing ties are resolved by node index.

5. Experimental results

5.1. Deadline-Satisfaction Comparison

Figure 2 compares six schedulers under balanced traffic. EDF-ECT exceeds ECT-FIFO in deadline satisfaction at loads 0.55, 0.75 and 0.95. At ρ=0.95, the means are 97.07% and 94.57%, a 2.50-percentage-point improvement over eight paired traces; the two-sided 95% Student-t interval is [1.73, 3.26] points. The matched random-number design holds every task potential service time on each worker fixed across policies, so this difference is attributable to routing and queue ordering within the replay.

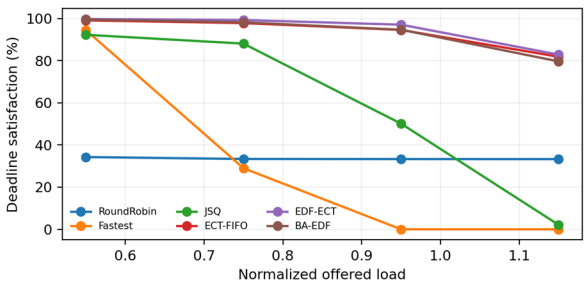


Fig. 2 Balanced-mix deadline-hit rate across eight arrival seeds (mean by load).

At ρ=1.15, balanced EDF-ECT attains 82.86% compared with 81.85% for FIFO, but the paired interval [−2.03, 4.07] percentage points crosses zero. Under large-heavy traffic the respective hit rates are 93.35% and 88.96%, with an interval [−0.35, 9.12]. Consequently, overload improvements cannot be distinguished reliably from replay-to-replay variation with eight seeds. The strongest evidence is at sub-overload settings, where the intervals on positive gains exclude zero.

Table II. Deadline-hit rate under both mixes (%)

Mix	ρ	ECT-FIFO	EDF-ECT	Δ pp
Balanced	0.55	99.12	99.76	+0.64
Balanced	0.75	97.83	99.28	+1.45
Balanced	0.95	94.57	97.07	+2.50
Balanced	1.15	81.85	82.86	+1.02
Heavy	0.55	99.62	99.94	+0.32
Heavy	0.75	99.01	99.82	+0.81
Heavy	0.95	96.91	99.27	+2.36
Heavy	1.15	88.96	93.35	+4.38

5.2. Queue Awareness Versus Static Placement

The fastest-profile baseline assigns many requests to the single-thread profile because it is usually fastest, yet that worker can become saturated while other profiles are idle. At ρ=0.95 under balanced traffic, Fastest has a 0% mean deadline-hit rate across the eight scored traces, despite a nominal aggregate capacity that could support more requests if work were distributed. RoundRobin shows about 33.33% completion by deadline at this load because it places one third of the stream on each node even when a profile is unable to sustain its assigned arrival rate. JSQ avoids some of this imbalance but treats a long and a short request as equivalent queue entries; its balanced-mix ρ=0.95 hit rate is 50.13%. ECT-based placement directly incorporates the profiled cost matrix and achieves substantially higher deadline compliance.

5.3. Response-Time Distribution and Trade-off

The deadline metric should be interpreted together with response-time tails. Figure 3 reports the balanced-mix 95th-percentile response time, averaged across seeds. At ρ=0.95, EDF-ECT has a p95 of 11.57 ms compared with 8.64 ms for ECT-FIFO. At ρ=0.75, the corresponding means are 7.90 and 6.42 ms. EDF promotes urgent jobs at the cost of leaving other requests waiting longer, which improves the number of requests meeting heterogeneous deadlines without necessarily reducing the unconditional high-percentile latency. This behavior is consistent with deadline scheduling objectives and tail-latency sensitivity [39], [41].

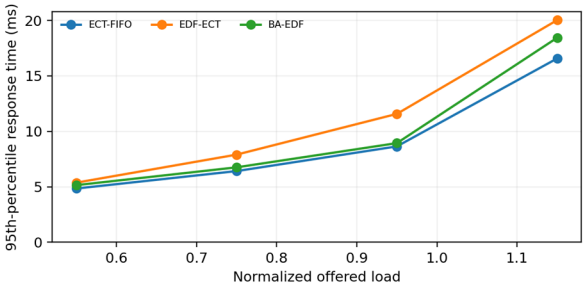


Fig. 3 Balanced-mix p95 latency of queue-aware schedulers, averaged over eight traces.

The bounded-EDF ablation does not uniformly resolve the trade-off. For balanced traffic at ρ=0.95, BA-EDF reaches 94.68% deadline satisfaction, compared with 94.57% for ECT-FIFO and 97.07% for unconstrained EDF-ECT. The limitation to two waiting jobs reduces reordering but can also forgo useful prioritization. The comparison suggests that an adaptive mode switch should consider lateness costs or

downstream deadline risk, not solely the number of queued jobs.

5.4. Sensitivity to Request-Class Composition

The large-heavy workload probes a distinct class mix while maintaining the same class-mix-adjusted normalized load definition. Since execution profiles and deadline allowances vary by class, identical ρ values do not imply identical request rates. At ρ=0.95, the arrival-rate reference is 867.5 requests/s in the balanced mix versus 850.0 requests/s for the large-heavy mix. Figure 4 reports deadline satisfaction: EDF-ECT attains 99.27% for large-heavy and 97.07% for balanced traffic, versus 96.92% and 94.57% for ECT-FIFO. These differences reflect the interactions of class deadlines, workloads, and dispatch rules, not a comparison of two equally sized requests.

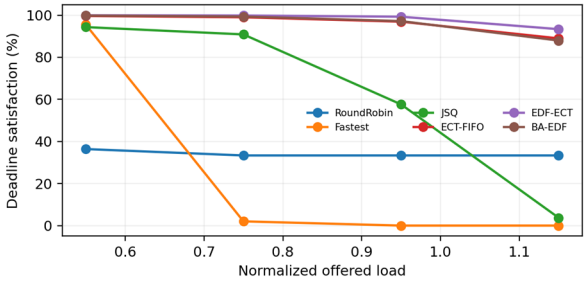


Fig. 4. Large-input-heavy deadline satisfaction, using the identical measured profiles.

At ρ=0.95 the paired EDF-ECT gain is 2.50 points [1.73, 3.26] for balanced and 2.36 [1.96, 2.75] for large-heavy traffic. Table III contrasts these improvements with increased p95 latency. The intervals characterize variation among eight generated demand traces given fixed empirical service distributions; they are not confidence bounds on actual networked edge-board performance. Common random numbers remove scheduler-dependent changes in which duration is sampled for a particular task/device pair.

Table III. Paired ect-fifo to edf-ect differences

Mix	ρ	Gain pp (95% CI)	Δ p95 ms
Balanced	0.95	+2.50 [+1.73, +3.26]	+2.93
Balanced	1.15	+1.02 [-2.03, +4.07]	+3.46
Heavy	0.95	+2.36 [+1.96, +2.75]	+2.01
Heavy	1.15	+4.38 [-0.35, +9.12]	+4.00

5.5. Illustrative Routing Decision

The profile matrix in Table I also explains the difference between queue-length and time-aware routing. Consider an illustrative small-class request arriving while virtual worker 1 has 4 ms of non-preemptive work remaining, worker 2 is idle, and worker 3 has one large-class job waiting. Ignoring transfer delays, the proposed finish-time estimates for a small incoming request are approximately 5.20 ms on worker 1, 3.76 ms on worker 2, and at least 14.37 ms on worker 3, calculated with the actual measured means. Thus ECT routes to worker 2 despite its higher standalone small-request execution time. Fastest would still route to worker 1, and JSQ would make a queue-length choice without evaluating this specific cost matrix. The example is an arithmetic illustration of the policy, not an additional empirical measurement.

Deadline ordering acts after routing and is relevant only when at least two requests wait for a worker. Assume a worker is executing a non-preemptive job, with a previously queued request having absolute deadline 48 ms and a newly arrived request having absolute deadline 42 ms. FIFO would start the older request first, whereas EDF starts the 42 ms request at the next release. EDF cannot affect the executing job. This distinction makes the algorithm implementable with ordinary per-worker priority queues and avoids the rollback or preemption mechanisms required by more elaborate real-time systems.

5.6. Statistical and Computational Accounting

The reported means and standard deviations are computed over eight replicates, not over individual requests pooled across seeds. For a pair of policies, each replicate uses identical arrivals and per-task/per-worker service samples, and the two-sided Student-t interval uses seven degrees of freedom. This approach prevents the thousands of highly dependent requests within a single trace from being misinterpreted as thousands of independent experimental replications. No search over tuning parameters was performed to select the reported results: class deadlines, load levels, seeds, and the two-job ablation threshold are explicitly specified in the executable experiment.

The replay cost consists primarily of simple deadline comparisons, in-memory queue scans, event ordering and fixed indexing into sampled service-time arrays. All profiles and policy variants fit into one script, and no external dataset download, model training or hardware deployment step is needed for a faithful rerun of this particular performance study. The experiment evaluates queue discipline rather than implementation overhead of a production network dispatcher. Consequently, the method is best interpreted as a low-complexity scheduling rule that is straightforward to transplant into a real edge controller once network and board measurements are collected.

6. Discussion

6.1. Interpretation for Edge Resource Management

The study supports a bounded scheduling claim: when inference times are profiled by task class and worker, combining completion-time placement with EDF queues can increase the rate of meeting class-specific deadlines under moderate load. This property does not depend on a particular pretrained model; it depends on how class-conditioned service costs, queue backlogs and deadlines interact. In a deployed edge cluster, a dispatcher could replace the profile matrix with timing measurements gathered separately on each board and add transfer delays to the finish-time estimate.

For multi-device deployment, additional variables must enter the objective: network uplink and downlink delay, input transfer size, branch placement, memory capacity, device background contention and energy. Model accuracy should be evaluated if inference paths or fidelity are modified. These aspects are not inferred from the present single-host data. Event-driven tools such as iFogSim and EdgeCloudSim similarly emphasize the need for explicit simulation assumptions [43-50].

6.2. Measurement Scope and Reproducibility

The 1/2/4-thread measurements are empirical CPU kernel observations, whereas queue interaction and parallel node independence are simulation assumptions. The measured

latency dispersion includes host runtime effects, and the markedly different mean performance at different thread settings is a property of this host, model and profiling session. Since all inferences execute an untrained graph on synthetic tensors, no classification accuracy, multimodal dataset benchmark or physical energy saving is reported. The code records all measured durations, seeds, queue disciplines and replay outputs. It can be rerun with separately obtained edge-board latency arrays without altering the algorithms [51-53].

In particular, the design does not presuppose that extra CPU threads provide faster kernels. Measurements show that larger thread pools may be slower for these small convolutions. Request-aware estimated finish-time placement is therefore more appropriate than assuming a monotonic ranking of profiles. MLPerf inference methodology provides a broader context for making inference measurements comparable, though this experiment is not an MLPerf submission [54-63].

7. Conclusion

This paper evaluated dynamic deadline-aware scheduling for dual-branch audio–visual requests using measured single-host service times and virtual heterogeneous workers. With matched request and service samples and workload-adjusted arrival rates, EDF-ECT improved deadline satisfaction from 94.57% to 97.07% at ρ=0.95 for balanced traffic, and from 96.92% to 99.27% for large-heavy traffic. The paired gains were 2.50 and 2.36 percentage points with 95% intervals excluding zero. EDF-ECT also increased unconditional p95 response time and showed uncertain gains under overload. These measured-profile replay results support an implementable request-level scheduling baseline and specify how it may be extended using network-aware physical-board profiles.

References

[1] Satyanarayanan, M., Bahl, V., Caceres, R., & Davies, N. (2009). The case for VM-based cloudlets in mobile computing. IEEE Pervasive Computing.

[2] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5), 637-646. https://doi.org/10.1109/JIOT.2016.2579198

[3] Mao, Y., You, C., Zhang, J., Huang, K., & Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. IEEE Communications Surveys & Tutorials, 19, 2322-2358. https://doi.org/10.1109/COMST.2017.2745201

[4] Fang, Q., & Liu, W. (2025). HARLA-ED: Resolving information asymmetry and enhancing algorithmic symmetry in intelligent educational assessment via hybrid reinforcement learning. Symmetry, 18(1), 58.

[5] Chen, T., Shang, W., & Guo, Z. (2024). XSLA: An SLA-aware heterogeneous XPU scheduling framework for edge AI inference. World Journal of Information Technology, 2(2), 48-59.

[6] Wang, G., Li, P., Feng, E., & Mo, T. (2024). Uncertainty-aware LLM-as-a-judge evaluation for reliable search and recommendation systems: Calibrated confidence and impact-aware selective human review. AI and Data Science Journal, 1(1), 112-121.

[7] Wang, J., Tan, Y., Jiang, B., Wu, B., & Liu, W. (2025). Dynamic marketing uplift modeling: A symmetry-preserving framework integrating causal forests with deep reinforcement learning for personalized intervention strategies. Symmetry, 17(4), 610.

[8] Yue, X., & Zhu, R. (2024). An explainable hybrid machine learning framework for cash-flow-conditioned multi-horizon liquidity risk early warning: An SME-oriented benchmark study. World Journal of Management Science, 2(1), 63-72.

[9] Chen, J., Han, X., Zhang, X., & Yang, Y. (2023). Exact concept-level explanations for early detection of financial misstatements. Eurasia Journal of Science and Technology, 5(2), 34-48.

[10] Zhou, F. (2026). Moment-constrained side-force allocation for wind-assisted merchant ships: Hydrodynamic performance of retractable anti-leeway fins and flap rudders. Innovation and Technology Studies, 3(1), 27-41.

[11] Ping, W., & Fan, H. (2023). Beyond isolated ratios: Accrual-revenue-cash flow indicators and interaction-aware learning for detecting financial reporting irregularities in U.S. public companies. World Journal of Economics and Business Research, 1(1), 46-55.

[12] Shen, A., & Liu, W. (2026). Budget-aware adaptive test-time compute with hierarchical self-verification for semantic ID generative recommendation. Computers.

[13] Yue, X., Yang, J., & Liu, W. (2026). FraudDebate-Agent: A multi-agent LLM framework with an evidence-based debate mechanism for financial statement fraud detection. Mathematics, 14(15), 2695.

[14] Zhang, F., & Yang, J. (2022). Accuracy-aware model compression for real-time visual perception under resource constraints. Journal of Computer Science and Electrical Engineering, 4(1), 45-52.

[15] Bian, M., Wang, Y., & Lin, Y. (2024). Explainable supplier risk prediction for critical telecommunications infrastructure procurement: An empirical XGBoost-SHAP framework. Social Science and Management, 1(1), 64-77.

[16] Liang, Y., Jiao, Y., Ping, W., Fan, H., & Han, X. (2026). Adaptive event-driven labeling: A neuro-symbolic multiagent framework for causal inference in non-stationary time series. IEEE Access.

[17] Shang, W., & Chen, T. (2024). PrefixKD-Edge: Real-time locomotion-intent prediction with prefix self-distillation on lightweight temporal networks. Innovation and Technology Studies, 1(1), 30-38.

[18] Chen, J., Liu, J., Liang, Y., & Zhou, M. (2026). HeteroGCL: A heterogeneous graph contrastive learning framework for scalable and sustainable cryptocurrency AML. Applied Sciences, 16(6), 2860.

[19] Qin, Y., Rhee, M., Teng, D., Zhang, C., & Zi, B. (2026). Neuro-symbolic constraint verification for LLM-driven internet finance transaction execution. Scientific Reports.

[20] Chen, J., Liang, Y., Liu, J., & Zhou, M. (2026). Temporal transformer with conditional tabular GAN for credit card fraud detection: A sequential deep learning approach. Mathematics, 14(7), 1183.

[21] Teng, D. (2025). TRIAD: Deadline- and reliability-aware scheduling for post-earthquake structural assessment workflows. World Journal of Engineering Research, 3(6), 55-66.

[22] Zhang, F., & Yang, J. (2024). Robustness-aware model compression for multimodal perception under challenging conditions. AI and Data Science Journal, 1(1), 103-111.

[23] Zou, J., Qin, Y., & Rhee, M. (2024). Beyond model scale: A task-aware reliability benchmark for large language models in clinical decision support. Innovation and Technology Studies, 1(1), 39-47.

[24] Chen, J., Zhang, X., & Han, X. (2023). Explainable identification of tax-related financial reporting anomalies in U.S. public companies: Cross-fitted tax-consistency residuals and Owen-value Shapley propagation. World Journal of Economics and Business Research, 1(1), 33-45.

[25] Liang, Y., & Jiao, Y. (2024). Real-time financial fraud detection using a feature-light temporal dynamic graph neural network with contrastive learning. Journal of Trends in Finance and Economics, 1(1), 76-84.

[26] Wang, J., Liu, J., Ge, Y., & Jiang, B. (2023). Risk-aware ensemble learning for credit default prediction under economic uncertainty. World Journal of Economics and Business Research, 1(1), 18-32.

[27] Wang, B., Zhao, W., & Wang, Z. (2024). DAS-GNN: A scalable disagreement-aware graph neural framework for anomaly detection in large-scale networks. AI and Data Science Journal, 1(1), 67-75.

[28] Teng, D., Rhee, M., Qin, Y., Zi, B., & Liu, W. (2026). SW-SpeedDLM: Sliding window speculative decoding for diffusion language models under long context constraints. Mathematics, 14(12), 2137.

[29] Cao, J., Zheng, W., Ge, Y., & Wang, J. (2025). DriftShield: Autonomous fraud detection via actor-critic reinforcement learning with dynamic feature reweighting. IEEE Open Journal of the Computer Society, 6, 1166-1177.

[30] Jiao, Y., & Liang, Y. (2024). An integrated process-mining and explainable machine-learning framework for bottleneck and internal-control risk detection in the multi-entity financial close. Social Science and Management, 1(3), 96-107.

[31] Guo, Z., Fang, Q., Yang, J. S., & Rhee, M. (2026). Closed-loop audio neuroregulation for ADHD: Graph-based attention decoding and rhythm entrainment. IEEE Access.

[32] Mai, N. T., Fang, Q., & Cao, W. (2025). Measuring student trust and over-reliance on AI tutors: Implications for STEM learning outcomes. International Journal of Social Sciences and English Literature, 9(12), 11-17.

[33] Wang, J., Liu, J., Zheng, W., & Ge, Y. (2025). Temporal heterogeneous graph contrastive learning for fraud detection in credit card transactions. IEEE Access.

[34] Zhu, R., & Yue, X. (2024). An explainable machine learning framework for predicting dividend sustainability and financial risk of US REITs under interest-rate shocks. Journal of Trends in Finance and Economics, 1(2), 48-57.

[35] Mai, N. T., Cao, W., & Fang, Q. (2025). A study on how LLMs (e.g., GPT-4, chatbots) are being integrated to support tutoring, essay feedback and content generation. Journal of Computing and Electronic Information Management, 18(3), 43-52.

[36] Qin, Y., Zou, J., & Rhee, M. (2022). An AI-driven intelligent transaction execution framework for internet finance platforms: Cost-sensitive real-time integrity interdiction. Eurasia Journal of Science and Technology, 4(2), 36-44.

[37] Wang, Y., & Zhang, C. (2023). AERIS: Adaptive executor-scoped resource management for privacy-preserving data processing in Apache Spark. Eurasia Journal of Science and Technology, 5(2), 25-33.

[38] Wang, G., Li, P., Feng, E., Shen, A., & Rhee, M. (2026). A dual-process memory-augmented large language model agent for explainable sequential recommendation. IEEE Access.

[39] Feng, E., Li, P., Mo, T., & Wang, G. (2022). PRISM: Prediction-guided risk-aware deep reinforcement learning for energy-aware dynamic resource scheduling in distributed cloud data centers. Journal of Computer Science and Electrical Engineering, 4(1), 53-61.

[40] Fan, H., Jiao, Y., Wang, M., Chen, J., & Yang, S. (2026). Fraud learns too: Continual graph learning under strategic adversarial drift in dynamic networks. Scientific Reports.

[41] Guo, Z., Chen, T., & Shang, W. (2022). Class-conditional intermediate-domain adversarial adaptation for cross-device acoustic scene classification. Journal of Computer Science and Electrical Engineering, 4(1), 26-35.

[42] Wang, Y., Bian, M., & Lin, Y. (2022). An explainable temporal graph neural network for disruption prediction and risk propagation in multi-tier supply chains. Journal of Computer Science and Electrical Engineering, 4(1), 17-25.

[43] Jiao, Y., Fan, H., Yue, X., Ping, W., Sun, T., & Wang, J. (2026). Dynamic heterogeneous graph contrastive learning for uncovering collusive financial fraud. Scientific Reports.

[44] Mo, T., Zhang, C., Zou, J., Guo, Z., & Rhee, M. (2026). Self-evolving AI agents with dual memory for automated software testing and bug localization. IEEE Access.

[45] Shen, A., & Huang, S. (2022). A systems-aware framework for intelligent large-scale recommendation models. Journal of Computer Science and Electrical Engineering, 4(1), 36-44.

[46] Zhang, F., Shen, A., & Huang, S. (2026). TRIAD: Co-designing mixed-precision quantization, operator fusion, and on-chip memory reuse for real-time vision-language-action inference on automotive NPUs. Results in Engineering, 113673.

[47] Rhee, M., Zou, J., & Qin, Y. (2024). CLARITY: A closed loop framework for duplicate aware failure prediction and budgeted automated remediation in cloud infrastructure. AI and Data Science Journal, 1(1), 76-85.

[48] Li, P., Feng, E., Wang, G., & Mo, T. (2024). Hybrid deep reinforcement learning and integer optimization for resource scheduling in heterogeneous AI computing clusters. Innovation and Technology Studies, 1(1), 48-56.

[49] Zhao, X., Liu, J., Wang, Y., & Wang, J. (2026). CryptoMamba-SSM: Linear complexity state space models for cryptocurrency volatility prediction. IEEE Open Journal of the Computer Society, 7, 226-243.

[50] Fang, Q. (2024). TRUE-EWS: Trustworthy early warning with equalized conformal guarantees for equitable student support in U.S. education. Educational Research and Human Development, 1(2), 44-54.

[51] Teng, D. (2025). RSRBench: Reproducibility and performance benchmarking for regional seismic risk workflows. Academic Journal of Architecture and Civil Engineering, 3(1), 40-52.

[52] Fang, Q. (2024). Bias-aware large language models for equitable educational feedback and assessment. Trends in Social Sciences and Humanities Research, 2(1), 41-50.

[53] Lin, Y., Yue, X., Bian, M., Zhu, R., & Teng, D. (2026). Subgraph-level temporal graph transformer with laundering-motif prototypes for anti-money laundering on blockchains. IEEE Access.

[54] Wang, J., Ge, Y., Liu, J., & Jiang, B. (2024). Early warning of liquidity stress in U.S. banks with seasonally aligned frailty hazards and delayed-feedback conformal alarms. Journal of Trends in Finance and Economics, 1(1), 85-100.

[55] Fan, H., Liang, Y., Jiao, Y., Ping, W., & Lin, Y. (2026). An explainable temporal machine learning framework for financial fraud risk prediction in listed institutions. Highlights in Business, Economics and Management, 70, 173-185.

[56] Zhang, C., & Wang, Y. (2024). A data-driven multi-objective production scheduling framework for battery pack manufacturing under material arrival uncertainty. AI and Data Science Journal, 1(1), 86-93.

[57] Mo, T., Wang, G., Feng, E., & Li, P. (2023). Execution-feedback-guided context selection for large language model-

based software testing and debugging. World Journal of Information and Knowledge Management, 1(1), 55-64.

[58] Lin, Y., Wang, Y., & Bian, M. (2024). XOR-EW: An explainable and cost-aware artificial intelligence framework for operational risk early warning in financial institutions. AI and Data Science Journal, 1(1), 94-102.

[59] Sun, T., Yang, J., Li, J., Chen, J., Liu, M., Fan, L., & Wang, X. (2024). Enhancing auto insurance risk evaluation with transformer and SHAP. IEEE Access, 12, 116546-116557.

[60] Zhang, H. (2024). Graph-aware multi-task learning for early prediction of timing violations and routing congestion in digital IC physical design. World Journal of Information Technology, 2(1), 13-20.

[61] Shen, A. (2023). Compute-efficient scaling of large sequential recommendation models under dynamic user behavior. World Journal of Information Technology, 1(1), 62-71.

[62] Chen, J., Cui, Y., Zhang, X., Yang, J., & Zhou, M. (2024). Temporal convolutional network for carbon tax projection: A data-driven approach. Applied Sciences, 14(20), 9213.

[63] Ping, W., & Fan, H. (2024). Disclosure quality and valuation risk in level 3 investments: A disclosure-implied risk framework for U.S. business development companies. Social Science and Management, 1(1), 78-87.