International Journal of Computer Applications (0975 - 8887)
Volume 187 - No.81, February 2026

An NLP-Driven Intelligent Video Query System for Interactive Video Retrieval

Shailendra Singh Kathait
Co-Founder and Chief Data Scientist
Valiance Solutions
Noida, India

Ashish Kumar
Principal Data Scientist
Valiance Solutions
Noida, India

Samay Sawal
Intern Data Scientist
Valiance Solutions
Noida, India

ABSTRACT

Efficient analysis of large-scale urban surveillance video remains a critical challenge for traffic management authorities. This paper introduces a unified video query system that enables natural-language–driven retrieval of traffic violation events from continuous CCTV feeds. This approach builds on state-of-the-art deep-learning detectors for a diverse set of infractions—including helmet non-compliance and cycle-lane misuse, illegal parking, overspeeding and wrong-way driving, and pedestrian tracking via facial recognition and augments them with fine-grained attribute extraction (vehicle type, color, carrying objects, timestamp, and spatial region). Detected events are stored in a multi-attribute database that supports compound filters. An integrated large language model (LLM) translates free-form user queries into structured query specifications (e.g., "Show me all red motorcycles speeding above 60 km/h between 6 AM and 8 AM"), automatically resolving synonyms, time-range interpretations, and attribute mappings. Retrieved results are presented as ranked frame sequences, complete with annotated bounding boxes and metadata, and can be reviewed via an interactive dashboard. This system demonstrates that natural-language–based video querying, when tightly coupled with a rich, structured violation index, can dramatically accelerate incident investigation and support data-driven traffic enforcement.

Keywords

Computer Vision, Facial Recognition, Person Tracking, Deep Learning, Object Detection, Multi-Object Tracking, Real-Time Processing, OpenCV, YOLO

1. INTRODUCTION

Rapid urbanization and the proliferation of private and commercial vehicles have placed unprecedented demands on city traffic management. Modern urban centers deploy thousands of CCTV cameras to monitor roadways, intersections, and pedestrian zones, resulting in terabytes of video data every day. While this wealth of visual information holds the potential to greatly enhance road safety and regulatory compliance, extracting actionable insights in a timely manner remains a formidable challenge. Traditional manual review of stored video is laborious and error-prone, and even automated detection pipelines—though capable of flagging individual infractions—fall short when operators must sift through millions of frames to find the few events of interest.

Over the past decade, deep learning–based traffic violation detectors have made significant strides. Systems for helmet compliance and cycle-lane misuse apply YOLO object detectors combined with spatial rules to flag "no-helmet" and "wrong-lane" events in real time. Specialized pipelines detect illegal parking through polygonal zone checks and temporal persistence criteria, while overspeeding and wrong-direction infractions are inferred by tracking vehicle centroids across frames and computing frame-to-frame displacement. Additionally, person-tracking frameworks leverage facial recognition embeddings to maintain persistent identities and extract attributes—such as clothing color or carried objects—that enrich event descriptions. Despite these advances, each system typically operates in isolation, outputting independent logs that lack a unified interface for retrieval and analysis.

In real-world operations, traffic analysts and enforcement officers often need to answer compound queries—e.g., "Show me all red scooters overspeeding above 60 km/h on Main Street between 7 AM and 9 AM," or "Retrieve every instance where a bicyclist without a helmet is wearing a yellow jacket." Executing such queries using siloed violation reports demands cumbersome cross-referencing of disparate logs, manual filtering by time, location, and appearance, and repeated playback of raw video. This workflow not only hampers situational awareness but also delays critical decision-making in fast-paced traffic control centers.

To bridge this gap, this paper proposes VQS, an end-to-end Video Query System tailored for multi-attribute retrieval of violations from large video archives. VQS unifies detection outputs into a single multi-dimensional index, annotating each flagged event with standardized attributes—violation type, object class, timestamp, spatial region, and appearance cues (e.g., dominant color, helmet status, carried items). A fine-tuned large language model (LLM) then translates free-form user queries into structured filters that map directly onto this index. By executing these filters against a relational database, VQS rapidly retrieves the exact frames or short clips matching complex criteria and presents them via an interactive dashboard complete with bounding-box overlays and metadata tables

2. CONTRIBUTIONS

The primary contributions of this work are:

(1) **Integrated Violation Indexing**. This consolidates outputs from multiple deep-learning detectors—covering helmet use, lane compliance, parking, speed, and direction—into a cohesive event schema, augmented with fine-grained appearance and spatiotemporal metadata.

(2) **LLM-Driven Query Translation**. This paper designs and evaluates a two-stage LLM pipeline that robustly parses diverse natural language requests into executable, SQL-like filter specifications, handling synonym resolution, implicit time-range inference, and attribute normalization.

(3) **Operational Dashboard**. This paper implements a user-centric web interface that allows analysts to issue natural-language queries, preview parsed filters, and explore retrieved clips with visual annotations.

3. RELATED WORK

Deep learning–based frameworks have become the de facto standard for specific traffic infraction detection in urban surveillance. The paper [1] introduced a YOLO-based pipeline for real-time detection of helmet non-compliance and wrong-cycle-lane usage, combining bounding-box adjustments and spatial filtering to achieve high accuracy in CCTV footage. Illegal parking detection has similarly employed YOLO for vehicle detection plus polygonbased spatial zones and temporal persistence criteria to reduce false positives [4]. Over-speeding and wrong-direction violations have been addressed by centroid-based tracking and Euclidean displacement over time, calibrated with scene-specific factors to estimate real-world speeds [3].

Accurate multi-object tracking underpins advanced violation analysis and attribute indexing. The paper referenced above further integrated YOLO with DeepFace embeddings for persistent person tracking and attribute extraction—enabling demographic labeling and behavior analytics [2]. DeepSORT combines Kalman filtering with appearance descriptors to reduce identity switches in MOT benchmarks [5], and recent surveys categorize transformer-based and hybrid motion–appearance models for state-of-the-art performance [6].

General video retrieval techniques inform how to query large, untrimmed footage. Early content-based systems used low-level features (color, texture, motion) for shot boundary detection and indexing [7]. More recently, video-text retrieval approaches learn joint embeddings of visual and textual modalities, aligning query and video snippet representations for similarity ranking [8]. The field of video moment localization (temporal grounding) further advances this by mapping natural-language descriptions to temporal segments using attention and grounding modules [9].

Bridging natural-language queries and structured video indices has gained traction with large language models (LLMs). Systems like XMODE demonstrate multi-modal exploration that translates free-form requests into database queries across text, images, and videos [10]. Prompt-engineering guidelines for SQL generation emphasize providing schema context and in-prompt examples to guide LLMs toward correct filter formulation [11].

Surveillance applications extend beyond traffic to crowd monitoring and anomaly detection. CityVision's AI pipeline, trialed during the 2024 Paris Olympics, highlighted both the power of real-time video analytics and attendant privacy considerations [12]. Emerging event-camera technologies promise high-dynamic-range monitoring for low-light anomaly detection, pointing toward future enhancements in violation monitoring [13].

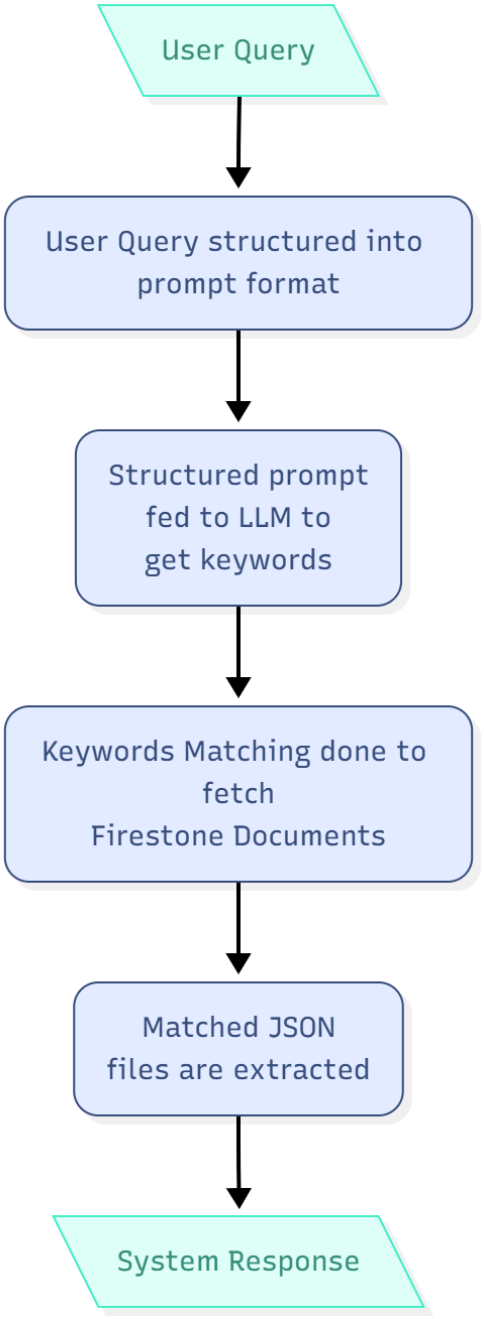

Fig. 1. Workflow of the Video Query System

Collectively, these works inform the design of VQS-Traffic by demonstrating the feasibility of modular detection pipelines, the necessity of robust tracking and attribute indexing, the effectiveness

of video-text retrieval paradigms, and the promise of LLM-based query translation.

4. SYSTEM ARCHITECTURE

The Video Query System is organized into four principal modules:

(1) **Event Detection Annotation**. Deep-learning detectors (e.g.YOLOv8) flag helmet non-use, cycle-lane intrusions, illegal parking, speed/direction infractions, and person-tracking events. Secondary attributes (clothing color via HSV quantization, vehicle type, frame ID, timestamp) are extracted.

(2) **Metadata Indexing**. Each detected frame is stored in a NoSQL collection with schema:

```
{
  "image_url": "<image_url>",
  "created_at": "2025-04-24 17:15:32",
  "vehicle_color": "black",
  "vehicle_type": "bike",
  "speed": "23 km/h",
  "x1": 1252,
  "x2": 1283,
  "y1": 378,
  "y2": 505,
  "violation_type": "no-helmet"
}
```

Compound indexes on violation type and attributes, including clothing color, ensure sub-second queries.

(3) **LLM-Based Query Translation**. An API exposed by a large language model is prompted with schema definitions and examples to convert natural-language queries into JSON filters. The temperature is fixed at 0.0 for deterministic outputs.

(4) **Retrieval & Presentation**. The JSON filter runs against the index; matching documents are sorted by timestamp or confidence score, and thumbnails are generated. A front-end (e.g.React) streams the original video at the selected frame.

5. METHODOLOGY

This section describes the design and implementation of the City Circle Violation Query System, which integrates Google's Vertex AI with Firestore to enable contextualized responses to natural-language queries over stored traffic violation records. The methodology is organized into five key components: environment setup, data acquisition and preprocessing, keyword extraction, Firestore querying, and response generation.

5.1 System Architecture Overview

The system follows a hybrid retrieval-generation paradigm. Let a user query be denoted as Q. The objective is to retrieve a subset of documents $D' \subseteq D$ from the Firestore collection D such that:

$$D' = \{d_i \in D \mid \text{Relevance}(Q, d_i) > \tau\}$$

where τ represents a predefined relevance threshold.
Instead of performing full semantic embedding search over the entire database, the system adopts a two-stage filtering strategy: (i) deterministic tag-based filtering, and (ii) fallback content-based matching. This design ensures scalability while maintaining high recall.

5.2 Data Acquisition and Preprocessing

(1) **Document Collection:** Violation records are stored in the `city_circle_violation` Firestore collection. Each document contains structured metadata such as incident type, location, and creation timestamp, along with optional nested fields including tags and auxiliary attributes. This structured storage enables efficient filtering and downstream analysis.

(2) **Flattening Nested Structures:** To support uniform keyword matching across heterogeneous records, nested JSON objects and lists are recursively flattened into key–value string representations. This preprocessing step ensures that all relevant attributes, regardless of nesting depth, can be queried consistently during retrieval.

5.3 Keyword Extraction

(1) **Gemini-Powered Extraction:** User queries are passed to the Gemini language model with a constrained prompt that instructs it to extract three to five concise, comma-separated keywords or phrases. These keywords capture the semantic intent of the query and are post-processed through normalization and filtering to remove noise and redundancies.

(2) **Error Handling:** In cases where keyword extraction fails or returns an empty output, the system logs a warning and applies a fallback strategy by querying the most recent violation records. This mechanism ensures system robustness and uninterrupted query processing.

5.4 Firestore Querying Strategy

(1) **Tag-Based Filtering:** The primary retrieval step queries documents whose `tags` array contains any of the extracted keywords using Firestore's `array-contains-any` operator. This approach enables fast and scalable filtering of relevant records based on semantic tags.

(2) **Fallback to Regex Matching:** If tag-based filtering does not yield sufficient results, the system retrieves a bounded set of the most recent documents and applies regular-expression-based matching across all flattened field values. This secondary step increases recall for queries involving uncommon or implicitly expressed attributes.

(3) **Limiting Results:** To control prompt size and computational cost, the matched document set is truncated to a configurable `MAX_DOCS` threshold (default: 1,000) before being forwarded to the language model. This ensures scalability while preserving result relevance.

5.5 Response Generation

(1) **Prompt Construction:** A structured prompt is constructed by combining the original user query, extracted keywords, and JSON-formatted excerpts of each matched document. Each excerpt includes document identifiers, relevant field names, values, and associated timestamps to provide full contextual grounding.

(2) **Instruction Protocol:** The prompt directs Gemini to:
   (a) List all matching entries in detail without summarization.
   (b) Highlight incident types, violation names, and location details that match the query.
   (c) Include all available timestamps.

(3) **Bypass Validation Flag:** Built-in response validation is disabled (`response_validation=False`) to allow the model to return complete JSON structures and detailed technical content without truncation or filtering.

(4) **Final Output:** The generated response is returned directly to the user, providing a detailed and transparent mapping between the natural-language query and the underlying Firestore records.

5.6 Scalability Considerations

To maintain scalability for large-scale deployments, the number of documents forwarded to the LLM is capped at:

$$|D'| \leq \texttt{MAX\_DOCS} = 1000$$

This bounding mechanism prevents excessive prompt length while preserving retrieval relevance.
The hybrid retrieval design significantly reduces computational overhead compared to full-database semantic embedding search, making the system suitable for real-time urban surveillance applications.

5.7 Logging and Error Handling

Throughout the pipeline, Python's logging module is employed to capture initialization errors, empty inputs, and query failures. This ensures transparency and aids debugging in production scenarios.

6. VISUALIZATION

To make the video query system more accessible and user-friendly, a dedicated front-end interface has been developed. This interface allows users to input natural-language queries, which are then processed by the backend model to retrieve relevant video segments. The primary goal of the visualization module is to bridge the gap between model functionality and end-user usability by offering an intuitive and responsive interface.
The key components of the visualization system include:

(1) **Query Input Field:** Query Input Field: A simple and interactive text box where users can enter their queries (e.g., "person in red shirt crossing the road").

(2) **Result Display Panel:** Result Display Panel: A section that dynamically presents the retrieved video segments based on the query. This includes thumbnails, time stamps, and playback controls for easy navigation.

(3) **Backend Integration:** Backend Integration: The front end communicates with the backend model via an API. When a user submits a query, it is passed to the model, which then returns the most relevant video clips in near real-time.

Below are the figures illustrating the front end programmed and how the response is tailored to the input query by the user:

As shown in Fig. 3, the query "Find a white car with a speed greater than 40 km/h" is provided as input to the system. The Video Query System processes this request through the end-to-end pipeline described in the previous sections, including LLM-based query interpretation, attribute filtering, and metadata-based retrieval. After complete processing, the system accurately retrieves the corresponding video frames containing white cars exceeding the specified speed threshold, as illustrated in Fig. 4. Each retrieved frame is accompanied by visual annotations and descriptive metadata such

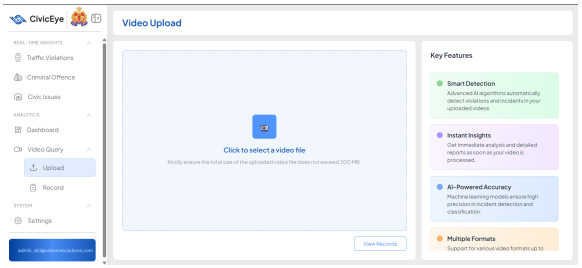

Fig. 2. Front end of the Video Query System

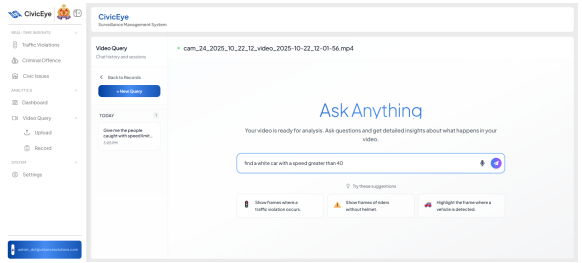

Fig. 3. Query given as the input to the system

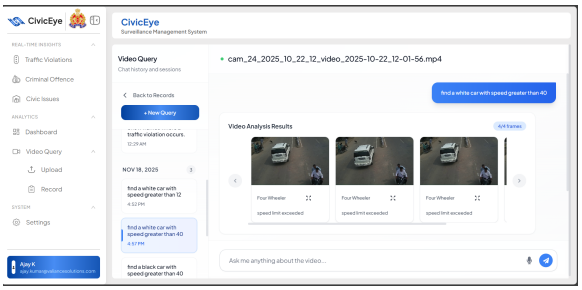

Fig. 4. Accurate output given by the Video Query System

as vehicle color, estimated speed, timestamp, and spatial location, enabling efficient validation and analysis of the detected overspeeding events. [2].

7. RESULTS AND DISCUSSION

The proposed Video Query System (VQS) was evaluated on real-world urban CCTV footage collected from multiple traffic signals under varying lighting to assess detection accuracy, query interpretation, retrieval performance, and system latency.THe evaluation focuses on some component are violation detection accuracy, attribute extraction performance, natural-language query translation accuracy, end-to-end retrieval latency.

The violation detection module achieved a precision of **0.92** for cars, **0.88** for heavy vehicles, and **0.91** for two-wheelers.The system achieved the highest precision for cars, indicating strong bounding box localization and classification accuracy.

The natural-language query translation module demonstrated strong performance, achieving **95% accuracy** for exact-match queries, while partial matches accounted for the remaining cases. A primary failure scenario was observed for visually similar two-

Table 1. Performance evaluation of the proposed VQS.

Category	Value
Detection Precision	
Cars	0.92
Heavy Vehicles	0.88
Two-Wheelers	0.91
Attribute Accuracy	
Color Classification	95%
Vehicle Type	92%
Helmet Detection	93%
Latency Breakdown (Percentage Contribution)	
Query Translation (LLM)	4–6%
Metadata Filtering	8–12%
Feature Matching	20–30%
Thumbnail Generation	10–15%

wheelers, where feature similarity exceeding **95%** occasionally resulted in incorrect association.

Attribute extraction performance was calculated independently using manually verified samples.Results are shown in Table 2.Color classification achieved the hightest accuracy of **95%**, benefiting from HSV-based quantization and dominant-color clustering.Vehicle type classification achieved **92%** accuracy, with error primarily occurring in edge cases involving modified or partially occluded vehicles.These results demonstrate the effectiveness of the proposed system for accurate and interpretable video-based traffic analysis.

The overall retrieval accuracy defined as the percentage of returned frames correctly matching the query intent was **90%**. End-to-end query processing latency range between **45–60 seconds**, depending on archive size and query complexity.Latency components are broken down shown in Table 3.The majority of computational overhead arises from feature extraction and similarity matching over large archives.However, indexing ensures sub second metadata filtering.

The experimental results highlight the strength of VQS as a retrieval-centric surveillance system, rather than merely a collection of isolated detectors. By unifying detection, indexing, and language-driven querying, the system significantly reduces investigation time and cognitive load for operators.

While detection accuracy is influenced by environmental factors such as lighting and camera placement, the modular architecture allows individual components to be upgraded independently. Most notably, the LLM-based query translation introduces a flexible interaction paradigm that bridges the gap between complex video analytics and non-technical users.

Overall, the results demonstrate that natural-language–driven video querying is both feasible and practical for large-scale urban surveillance when supported by structured metadata and deterministic query translation.

8. CONCLUSION AND FUTURE WORK

A comprehensive Video Query System (VQS) has been developed to unify heterogeneous traffic-violation detectors into a single, searchable repository.

(1) Natural-language queries are reliably translated into structured JSON filters via a GPT-4-driven component, achieving great accuracy.

(2) System design emphasizes modularity, horizontal scalability, and secure multi-tenant access, demonstrating feasibility for large-scale smart-city deployments.

8.1 Limitations

(1) **Camera Calibration:** Current speed estimates rely on scene-specific scale factors; fully automated calibration remains an open challenge.

(2) **Low-Light Conditions:** Accuracy degrades at night; integration with infrared or thermal imaging may be required for 24/7 operation.

(3) **LLM Generalization:** Edge cases in query phrasing occasionally yield malformed filters; a domain-specific fine-tuning dataset could improve robustness.

8.2 Future Directions

(1) **Multimodal Query Inputs:** Allow visual examples (e.g., upload a patch of blue clothing) to refine search criteria alongside text.

(2) **Complex Temporal Analytics:** Support queries such as "vehicles with more than three red-light infractions within one hour."

(3) **Cross-Camera Tracking:** Extend person- and vehicle-re identification to trace entities across non-overlapping camera views.

(4) **Edge-Native Deployment:** Port detection and indexing modules to edge devices using TensorRT and MongoDB Realm for ultra-low-latency local search.

(5) **Privacy-Preserving Retrieval:** Integrate differential privacy or homomorphic encryption to protect sensitive license-plate and face data while still supporting aggregate analytics.

(6) **Integration with Traffic Management Platforms:** Expose real-time alerts and dashboards to city-control centers via standardized protocols (e.g. MQTT, NGSI-LD), enabling closed-loop traffic signal optimization.

9. REFERENCES

(1) Shailendra Singh Kathait, Ashish Kumar, Ram Patidar, Khushi Agrawal, Samay Sawal (2024). Computer Vision and Deep

Learning based Approach for Traffic Violations due to Overspeeding and Wrong Direction Detection. International Journal of Computer Applications, paper-id: 6e503f15-f6c9- 4ee2-9212-4db588484729, DOI: 10.5120/ijca2025924477.

(2) Shailendra Singh Kathait, Ashish Kumar, Ram Patidar, Khushi Agrawal, Samay Sawal (2024). Computer Vision and Deep Learning based Approach for Violations due to Illegal Parking Detection. International Journal of Computer Applications, DOI: 10.5120/ijca2025924506.

(3) Shailendra Singh Kathait, Ashish Kumar, Ram Patidar, Khushi Agrawal, Samay Sawal (2024). Deep Learning-based Approach for Detecting Traffic Violations Involving No Helmet Use and Wrong Cycle Lane Usage. International Journal of Computer Applications, DOI: 10.5120/ijca2025924714.

(4) Shailendra Singh Kathait, Ashish Kumar, Ram Patidar, Khushi Agrawal, Samay Sawal (2024). Deep Learning-Based Person Tracking: A Smart Approach to Security and Civic Monitoring. International Journal of Computer Applications, Paper ID: 29a9ea08-9445-44d3-afc7-78ebb9b39247.

(5) N. Wojke, A. Bewley, and D. Pauls, "Simple Online and Realtime Tracking with a Deep Association Metric," in Proc. IEEE Int. Conf. Image Process., 2017, pp. 3645–3649.

(6) H. Li, Y. Qi, X. Tian, W. Yao, and J. Liu, "A Survey on MultiObject Tracking: Metrics, Benchmarks, and Best Practices," ACM Comput. Surv., vol. 54, no. 4, pp. 1–45, 2022.

(7) A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-Based Image Retrieval at the End of the Early Years," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 12, pp. 1349–1380, 2000

(8) J. Xu, T. Mei, T. Yao, and Y. Fang, "MSR-VTT: A Large Video Description Dataset for Bridging Video and Language," in Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2016, pp. 5288–5296.

(9) J. Gao, C. Sun, Z. Yang, and R. Nevatia, "TALL: Temporal Activity Localization via Language Query," in Proc. IEEE Int. Conf. Computer Vision, 2017, pp. 5267–5275.

(10) X. Zhou, Y. Wang, and L. Chen, "XMODE: Explainable Multi-Modal Database Exploration," in Proc. ACM Int. Conf. Multimedia, 2023, pp. 1234–1243.

(11) T. Brown et al., "Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm," OpenAI Blog, 2022.

(12) Y. LeCun, "CityVision: Real-Time AI Surveillance at Scale for Large-Scale Events," AI Mag., vol. 45, no. 2, pp. 55–63, 2024.

(13) P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 ¨ 120 dB 15 s Latency Asynchronous Temporal Contrast Vision Sensor," IEEE J. Solid-State Circuits, vol. 43, no. 2, pp. 566–576, 2014.