TDLight: A Framework for Incremental Light-Curve Management and Continuous Classification Optimization

Xinghang Yu [1,2] Ce Yu [1,2] Zeguang Shao [1,2] Chen Wang [3] and Bin Yang [1,2]

[1] School of Computer Science and Technology, Tianjin University, 135 Yaguan Road, Haihe Education Park, Tianjin 300350, People's Republic of China

[2] Technical R&D Innovation Center, National Astronomical Data Center, 135 Yaguan Road, Haihe Education Park, Tianjin 300350, People's Republic of China

[3] College of Computing and Data Science, Nanyang Technological University, 50 Nanyang Ave, Block N 4, Singapore 639798

ABSTRACT

Time-series observations of celestial objects are fundamental to studying variable and transient phenomena. As time-domain surveys continue to scale up, timely automated classification is essential for rapid follow-up. Many existing approaches treat storage and analysis as separate stages, archiving data first and analyzing it later. Yet each source is a continuously growing stream, and early observations often contain sufficient information to assign scientifically useful labels before the full light curve is available. Motivated by this, we present TDLight, a framework for light-curve management and trigger-based classification over incremental data streams. Built on the time-series database TDengine, the system loads the *Gaia* DR2 archive ($\sim 4.8 \times 10^7$ rows) in 33s and supports low-latency retrieval. As new observations accumulate, TDLight automatically triggers a lightweight LightGBM-based classification pipeline that requires no GPU. When prediction confidence exceeds a predefined threshold, these early results are written back to the database, providing reliable classifications before offline analysis is complete. The deployed system also supports manual label confirmation and optional incremental training. Evaluated on the *LEAVES* dataset ($\sim$1.10 million sources spanning 10 classes), the system achieves an overall accuracy of 93.6% and a weighted F1 score of 93.7%. Furthermore, it identifies 560 high-confidence catalog-disagreement candidates for follow-up verification. Notably, this includes correcting 150 RRc variables erroneously labeled as EW binaries due to inherited period-doubling artifacts. TDLight and the related data products are publicly available.

Keywords: Astroinformatics (78) — Astronomical databases (83) — Time domain astronomy (2109) — Light curves (918) — Classification (1907) — Variable stars (1761)

1. INTRODUCTION

Time-domain astronomy studies the temporal evolution of celestial objects through their light curves, which encode critical information about phenomena ranging from stellar pulsations to explosive transients. The identification and classification of variable sources from these light curves is fundamental to modern astrophysics. Certain classes of variable stars, particularly Cepheids and RR Lyrae stars, serve as important distance indicators and provide key constraints on stellar structure and evolution (A. G. Riess et al. 2016; G. Clementini et al. 2019). Observations of Type Ia supernovae provided key evidence for the accelerated expansion of the Universe (A. G. Riess et al. 1998; S. Perlmutter et al. 1999), and the discovery of electromagnetic counterparts to gravitational waves has underscored the need for rapid analysis and follow-up (B. P. Abbott et al. 2017). Across all these domains, scientific yield is maximized when targets are accurately characterized at early observational stages, making timely and automated classification a central requirement.

The landscape of time-domain surveys has scaled dramatically, from early projects such as MACHO (C. Alcock et al. 2000) and OGLE (A. Udalski et al. 2015) to modern high-cadence facilities including the Zwicky Transient Facility (ZTF; E. C. Bellm et al. 2019) and *Gaia* (Gaia Collaboration et al. 2023). The next-generation Vera C. Rubin Observatory Legacy Survey of Space and Time (LSST) will extend this trend further, making automated classification

Corresponding author: Bin Yang
yangbincic@tju.edu.cn

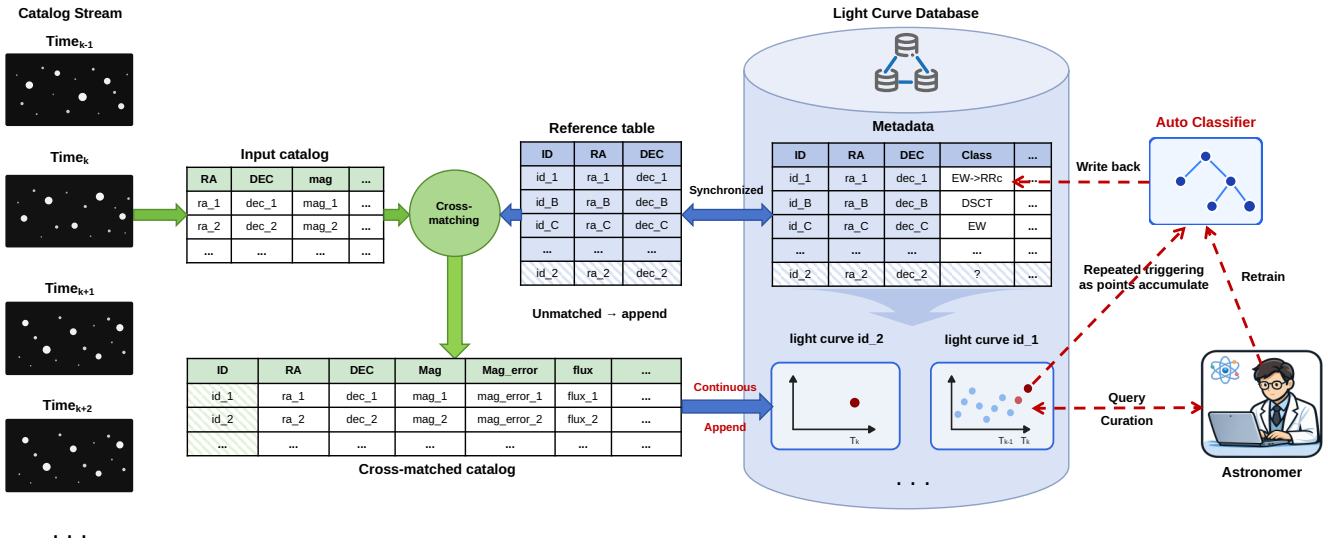

Figure 1. Overview of the TDLIGHT workflow for incrementally arriving catalogue streams. New catalogues are cross-matched against a reference table and appended to a source-centered light-curve database. As each source accumulates new observations, the system re-triggers feature extraction and classification, and writes high-confidence results back to the database before the full light curve becomes available.

and anomaly detection indispensable (Ž. Ivezić et al. 2019; C. J. Fluke & C. Jacobs 2020). Accordingly, a broad range of machine-learning methods has been developed for light-curve analysis, from traditional feature-based classifiers to recent deep-learning models, including foundation-model approaches (C. Yu et al. 2021; J. W. Richards et al. 2011; D.-W. Kim et al. 2014; A. D'Isanto et al. 2016; X. Zuo et al. 2026). In practice, however, these methods are still most often applied in an offline manner after data collection, rather than being embedded into the data-management workflow. This separation is increasingly limiting for incrementally growing light curves, whose classifications may need to be revisited as new observations arrive. Moreover, the computational and engineering demands of many existing models make frequent, low-latency re-analysis difficult in lightweight local deployments. As a result, classification is commonly treated as a post-processing task, decoupled from the primary storage and update loop.

Existing approaches fall into two broad categories. *Archival systems* are optimized for time-series storage and retrieval: for example, LCGCT (Z. Zhang et al. 2024) uses a purpose-built time-series database to replace a traditional relational backend, achieving substantial storage savings and faster photometric queries; however, scientific analysis in such systems is still typically performed after data retrieval. *Real-time alert brokers*, such as ALeRCE (F. Förster et al. 2021) and the broader ZTF alert infrastructure (M. T. Patterson et al. 2019; F. J. Masci et al. 2019), are designed for community-scale alert distribution and typically separate real-time event handling from long-term source-centered local storage. In practice, many light curves are formed progressively as catalogues are cross-matched over time, and neither category supports repeated source-level analysis during data accumulation. What is still lacking is a lightweight database-centered framework that supports repeated source-level classification and confidence-controlled write-back as new observations accumulate.

We present TDLIGHT, a framework that unifies ingestion, reclassification, and database write-back within a single source-centered workflow, embedding classification directly into the data path rather than treating it as a separate offline task. Built on TDENGINE[4], the system manages each source as an independent time-series entity with HEALPix

spatial indexing, supporting high-throughput incremental ingestion and sub-second retrieval. When sufficient new observations have accumulated, TDLIGHT automatically re-extracts features and runs a LightGBM-based hierarchical classifier, writing results back to the database only when the prediction confidence exceeds a predefined threshold and deferring uncertain cases until more data arrive. As evidence progressively accumulates, the features become increasingly representative, allowing reliable results to be accepted before the full light curve is available.

Beyond its software architecture, TDLIGHT serves as a powerful tool for catalog auditing and data curation. By deploying the framework on the multi-survey *LEAVES* dataset, we identified 560 high-confidence misclassifications in existing archives. Most notably, we discovered and corrected 150 RRc variables that had been systematically misclassified as W Ursae Majoris (EW) binaries due to inherited period-doubling artifacts. Along with the lightweight, GPU-free open-source software, we release a curated supplementary catalog of these corrected sources to the community.

This paper is organized as follows. Section 2 describes the two datasets used in this study. Section 3 introduces the overall TDLIGHT framework, including the system architecture, data model, and data flow. Section 4 evaluates the framework in terms of incremental ingestion, spatial query efficiency, hierarchical classification performance on LEAVES, early classification enabled by trigger-based updates, and the discovery of systematic catalog disagreements. Section 5 describes the deployment environment and web interface. Section 6 concludes with a summary and prospects for future work.

2. DATA PREPARATION

We use two datasets with complementary roles in the experiments. The *Gaia* DR2 variable-star archive provides the large-scale time-series workload for ingestion and retrieval benchmarks, while the LEAVES catalog provides labeled samples for classification evaluation. Table 1 summarizes both datasets.

Table 1. Summary of the two datasets used in this work.

	Gaia DR2	LEAVES
Sources	549 370	∼1.10 M
Photometric records	$\sim 4.8 \times 10^7$	$\sim 3.8 \times 10^8$
Classes	—	10 leaf classes
Role	Ingestion & query benchmark	Classification training & evaluation

2.1. *Gaia DR2 Light-Curve Archive*

We retrieved epoch photometry for all 549,370 confirmed variable sources from the *Gaia* Data Release 2 (DR2) variability tables, using the public *Gaia* archive (Gaia Collaboration et al. 2018; B. Holl et al. 2018). The Gaia DR2 data used in this work are available from the ESA Gaia Archive(G. Collaboration 2018). The resulting dataset contains approximately 4.8×10^7 individual photometric measurements in the G, $G_{\rm BP}$, and $G_{\rm RP}$ bands, whose photometric content and calibration are described by D. W. Evans et al. (2018). Because the variability tables do not directly provide the astrometric positions required by our storage and spatial-query pipeline, we joined them with the main *Gaia* DR2 source catalog through `source_id` to obtain right ascension and declination, and then converted these coordinates into HEALPix indices ($N_{\rm side} = 64$) for spatial indexing (Section 3).

We adopted DR2 in this work because it already provides a sufficiently large and realistic benchmark for evaluating the ingestion and indexing behavior of TDLight under large-scale workloads. In addition, the DR2 data can be reorganized in a straightforward and reproducible way into the source-level and epoch-level layouts required by our experiments. While *Gaia* DR3 provides expanded variability products, including epoch photometry for many more sources, these data are distributed through the archive's DataLink products rather than simple table queries, and incorporating them into the same benchmark would require additional data preparation. We therefore leave extension to later *Gaia* releases to future work.

To evaluate both ingestion modes of TDLight, we reorganized the raw data into two complementary layouts: (i) 549,370 per-source CSV files, each containing the complete multi-band light curve of a single object, for the archival

[4] https://www.tdengine.com

(batch) benchmark; and (ii) 728 epoch-style catalog files, each containing a single-epoch snapshot of many sources, to simulate streaming alert ingestion.

2.2. *LEAVES Classification Dataset*

For classification, we adopt the LEAVES dataset (Y. Fei et al. 2024), a multi-survey variability catalog drawn from ASAS-SN, ZTF, and *Gaia*. The LEAVES dataset used in this work is available (C. Yu 2024). The underlying survey resources include ASAS-SN light curves (C. S. Kochanek et al. 2017; T. Jayasinghe et al. 2018, 2019; C. T. Christy et al. 2023), ZTF light curves hosted by IPAC (ZTF Team 2025) and *Gaia* DR3 data products (G. Collaboration 2022). The dataset contains ∼1.10 million labeled sources organized into 10 leaf classes under a four-level hierarchical taxonomy. Table 2 lists the per-class counts; EW and SR together account for over 60% of the sample, while CEP is the rarest class. We use the published labels and light curves without modification.

Table 2. Class distribution of the LEAVES 10-class dataset.

Class	Count	Class	Count
EW	336 875	RRAB	41 823
SR	331 111	M	19 168
Non-var	134 592	RRc	18 703
ROT	119 112	DSCT	18 600
EA	77 356	CEP	2 455

For each source, we extract 15 statistical features from its light curve using the FEETS package (J. B. Cabral et al. 2018), including the Lomb–Scargle period (N. R. Lomb 1976; J. D. Scargle 1982), Stetson K index, amplitude, and skewness. These features are used as input to the hierarchical LightGBM classifier described in Section 4.3.

3. TDLIGHT FRAMEWORK

3.1. *Architectural Overview*

The TDLIGHT framework builds on TDENGINE, a database originally designed for Internet of Things (IoT) telemetry, and adapts it to astronomical time-domain data. Because light curves are inherently time-ordered, a time-series database is a natural backend for their storage and incremental management. In this design, each astronomical source is treated as an independent time-series entity, while TDENGINE's columnar architecture and native support for time-series workloads provide an efficient basis for append-oriented ingestion, compression, and source-centered data management.

Data model. TDLIGHT adopts a "one-table-per-source" schema, in which each source is stored as an individual TDENGINE *child table* beneath a shared *supertable*. This layout matches both TDENGINE's native data model and the source-centered nature of light-curve data. Observations of the same source remain grouped in time order, while static attributes such as coordinates, HEALPix indices, and class tags are stored once as tag columns, making the design well suited to incremental ingestion and per-source retrieval.

For comparison, a conventional relational design stores all sources in a single shared table indexed by `source_id` and timestamp. Although flexible, this layout is less naturally matched to the source-centered incremental workflow of TDLIGHT, in which observations are appended over time and classification results are repeatedly updated at the source level. As shown in Table 3, the per-source schema provides higher ingestion throughput and lower storage overhead than the relational baselines under this source-centered workload. Overall, this behavior is consistent with the design goals of TDLIGHT.

Figure 2 summarizes the overall architecture of TDLIGHT, which is organized into three coordinated layers: storage, software, and user interface.

The **Storage Layer** manages both photometric time-series data and static source metadata in TDENGINE. Time-varying observations are stored in per-source child tables, while coordinates, HEALPix indices, and classification tags are recorded as tag columns.

The **Software Layer** contains three main modules: data ingestion, data retrieval, and classification. The ingestion module parses external files and inserts records into the database; the retrieval module translates user queries into

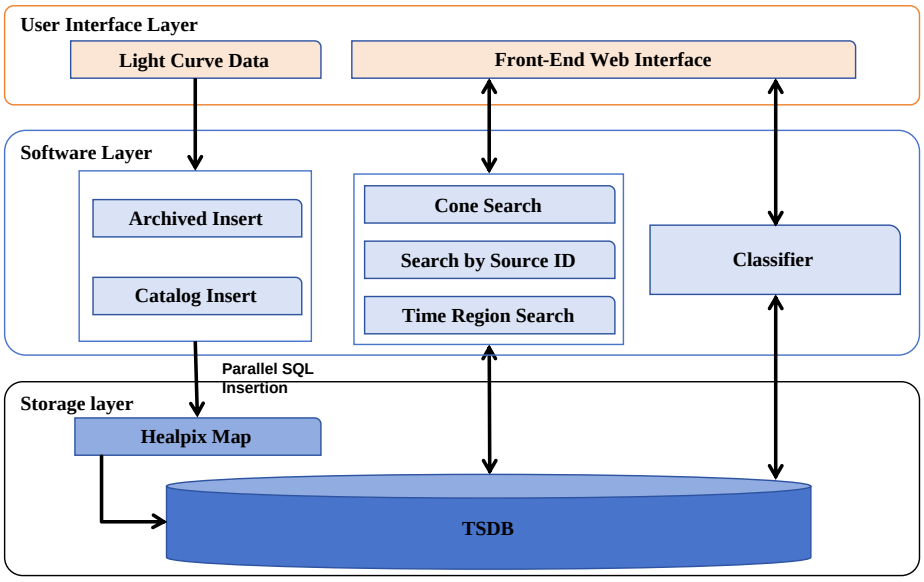

Figure 2. Overview of the TDLIGHT framework, including the user interface, software, and storage layers.

source-level and spatial database operations; and the classification module is triggered after data updates, performing feature extraction, LightGBM-based inference, and high-confidence result write-back.

The **User Interface Layer** is a web-based frontend through which users inspect light curves, query sources, and view classification results. It communicates with the Software Layer through REST APIs rather than accessing the database directly.

3.2. *Data Flow*

TDLIGHT supports two complementary ingestion modes. In archival mode, as in the *Gaia* DR2 benchmark, each source is stored as an individual CSV file and ingested into its corresponding per-source table. In streaming mode, each incoming catalog contains a single-epoch snapshot of many sources. To maintain source continuity across epochs, these records are first associated with existing sources through a reference table, following the reference-table-based source association idea of ASTROCATR (C. Yu et al. 2020) and the broader literature on astronomical cross-identification (T. Budavári & A. S. Szalay 2008). The reference table stores source identifiers together with positional metadata and is used—through HEALPix-based candidate filtering followed by sky-position matching—to determine whether an incoming record should be matched to an existing source or initialized as a new one. Matched records are appended to the corresponding per-source tables, while unmatched records are inserted with newly created source metadata. In both modes, HEALPix indices are computed from the source coordinates and recorded as metadata for subsequent spatial queries.

After new observations are inserted, the system schedules feature extraction and LightGBM-based classification for newly ingested sources as well as previously seen sources whose light curves have grown sufficiently since the last update. When the prediction confidence exceeds a predefined threshold, the classification result is written back to the database as a tag update; otherwise, the source remains unlabeled until more observations become available.

Users can interactively query the database through the web frontend using cone searches, source ID lookups, or rectangular sky regions, and inspect both the accumulated light curves and the current classification results. In this way, ingestion, cross-matching, storage, repeated classification, and result write-back are integrated into the same workflow.

4. EXPERIMENTS AND EVALUATION

We benchmarked system performance using a dataset derived from *Gaia* DR2, comprising 549,370 unique objects and ∼48.2 million photometric records. The data were reorganized into two formats: (i) 549,370 individual per-source CSV files for archival loading, and (ii) 728 epoch-style catalog files—each containing a single-epoch snapshot of multiple

sources—for simulated alert-stream ingestion. All benchmarks were conducted on a dual-socket Intel Xeon Gold 6530 server (64 threads, NVMe SSD) running TDengine 3.4.

4.1. *Ingestion*

Archival Ingestion (Batch Mode). In this mode, 549,370 per-source light curve files are loaded in parallel. Source coordinates are precomputed into HEALPix indices ($N_{\rm side} = 64$) and cached locally. Concurrent worker threads execute batched insertions, leveraging TDengine's automatic table creation to eliminate pre-scanning overhead. As shown in the right panel of Figure 3, this pipeline achieves a peak end-to-end throughput of 1.48×10^6 rows s^{-1} at 32 threads, completing the full 48.2-million-row ingest in 33 s.

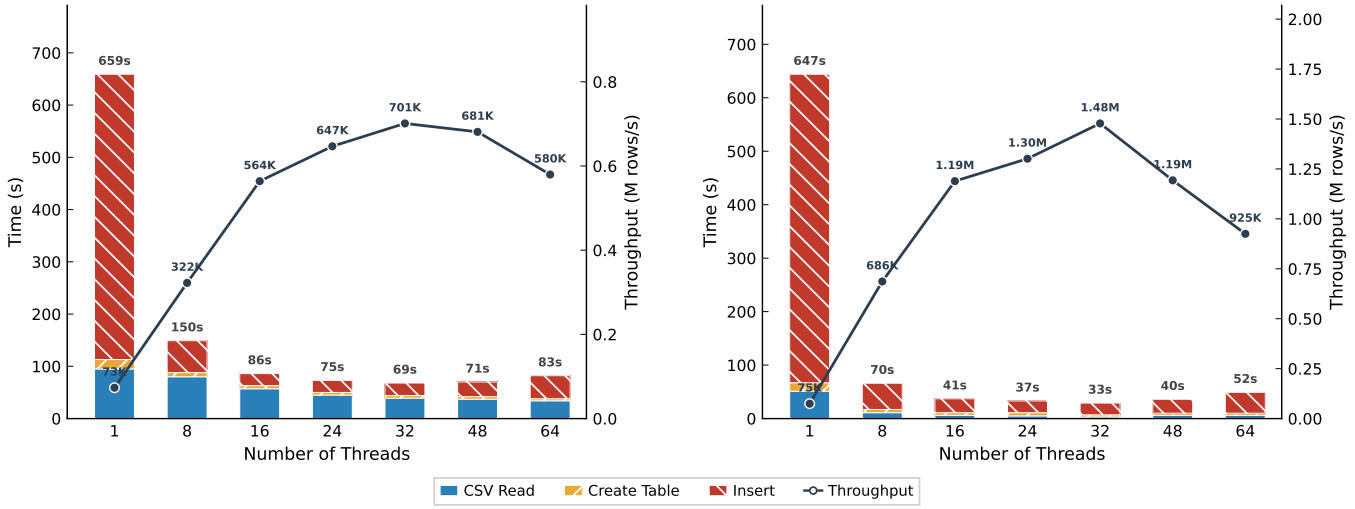

Figure 3. Ingestion performance on the *Gaia* DR2 benchmark. Left: streaming mode, peaking at 7.0×10^5 rows s^{-1}. Right: archival mode, peaking at 1.48×10^6 rows s^{-1}.

Streaming Ingestion (Real-time Mode). To simulate alert streams, epoch-style catalogs are ingested sequentially: each catalog is parsed, and individual records are routed to the corresponding per-source tables via HEALPix mapping. To minimize database round-trips, the ingestion engine aggregates insertion statements targeting multiple child tables into unified batch requests. The left panel of Figure 3 shows that this strategy sustains a peak throughput of 7.0×10^5 rows s^{-1} at 32 threads.

Both modes exhibit a throughput peak near 32 threads followed by a slight decrease at higher thread counts, which we attribute to lock contention within the storage engine when the number of concurrent writers exceeds the number of physical cores per socket.

Comparison with alternative database backends. Figure 3 shows the end-to-end ingestion performance of TDLight on the full archival benchmark described above. For comparison, we benchmarked the same framework with three relational backends—PostgreSQL[5], TimescaleDB[6], and SQLite[7], using the same benchmark workload, a unified C++17 implementation, and identical CSV parsing, HEALPix computation, and timing logic. For fairness, TDengine, PostgreSQL, and TimescaleDB were driven by 32 client threads, whereas SQLite was evaluated with a single writer thread because its write path remains globally serialized and additional threads degraded performance. Table 3 summarizes the comparison.

With the TDengine backend, the framework achieves the highest ingestion throughput, reaching 1.63×10^6 rows s^{-1}, compared with 6.98×10^5 rows s^{-1} for PostgreSQL, 3.43×10^5 rows s^{-1} for TimescaleDB, and 1.10×10^6 rows s^{-1} for SQLite. It also yields the smallest on-disk footprint (4.4 GB, versus 8.5 GB for PostgreSQL and TimescaleDB, and 6.8 GB for SQLite). PostgreSQL and TimescaleDB show broadly similar behavior in this workload, with TimescaleDB incurring a noticeable reduction in ingestion throughput because its hypertable chunk routing adds overhead during

[5] https://www.postgresql.org/docs/
[6] https://docs.timescale.com/self-hosted/latest/
[7] https://www.sqlite.org/

bulk insertion. SQLite achieves surprisingly high throughput despite running with a single writer thread, reflecting its low embedded overhead; however, it is included primarily as a lightweight local baseline rather than as a head-to-head competitor to the server-grade backends. Overall, these results indicate that the per-source schema enabled by TDengine is well suited to high-throughput archival ingestion under the source-centered workload of TDLight, while the relational baselines provide lower write performance and larger storage footprints under the same schema constraints. This combination of higher ingestion throughput and a more compact archive footprint is particularly attractive for downstream light-curve services in survey environments approaching the scale of Rubin Observatory's Legacy Survey of Space and Time (LSST; Ž. Ivezić et al. 2019). Current Rubin project materials indicate a nightly data volume of about 10 TB and an alert stream of order 10^7 events per night.[8] Although our benchmark addresses source-level photometric ingestion rather than raw image handling, the measured results suggest that TDLight can serve as an efficient archive layer for such time-domain workloads.

Table 3. Archival-mode performance with per-source and relational schemas. All results are obtained from the same C++17 benchmark framework on the same server. Throughput is computed from pure database time excluding CSV parsing. For the relational backends, no secondary indexes are created on the single-table schema; SQLite is evaluated with one writer thread because concurrent writes are serialized by its locking model, whereas PostgreSQL, TimescaleDB, and TDLight use 32 threads. Cone-search latencies are measured with 500 random queries (0.5° radius) that perform direct *ra/dec* spatial filtering via SQL.

	SQLite	PostgreSQL	TimescaleDB	TDLight
Threads	1	32	32	32
Write Throughput (rows s^{-1})	1.10 M	698 k	343 k	1.63 M
Cone-search med. (ms)	2152	1089	1070	86.44
Cone-search P90 (ms)	2383	1172	1300	131.95
Cone-search P99 (ms)	2464	1215	1858	337.06
Disk (GB)	6.8	8.5	8.5	4.4

4.2. *Spatial Query*

For cone searches, TDLight adopts a HEALPix-based filter-and-refine strategy. HEALPix-based metadata filtering is first used to restrict the search to a small set of relevant sky pixels, after which an in-memory geometric refinement is applied to the remaining candidates. In the TDLight implementation, this strategy reduces cone-search latency from about 10 s for a full-table scan to 0.10–0.38 s over search radii from $1'$ to $20'$, corresponding to speedups of approximately 27× to 104×.

To assess cross-backend cone-search performance under identical spatial semantics, we benchmarked 500 random queries (0.5° radius) using a unified C++17 framework. All backends execute the same cone-search logic: a rectangular *ra/dec* bounding-box pre-filter followed by an exact angular-distance test.

Table 3 summarizes the results. Under this spatial-filter workload, the TDLight storage layout achieves a median latency of 86 ms, compared with 1.07 s for TimescaleDB, 1.09 s for PostgreSQL, and 2.15 s for SQLite. The difference reflects the design-level mismatch between the source-centered spatial-query pattern and the conventional single-table relational schema: TDLight resolves each cone to a small set of HEALPix pixels and queries only the corresponding per-source subtables, whereas the relational backends must evaluate the same spatial predicate across the entire monolithic table without a spatial index. TimescaleDB and PostgreSQL show broadly similar median latency, with TimescaleDB exhibiting a larger tail (P99 = 1.86 s versus 1.22 s for PostgreSQL) because its hypertable chunk routing adds overhead without benefit when no time-based or spatial pruning is possible. The SQLite result (2.15 s median) is included as a lightweight embedded reference rather than as a head-to-head competitor.

These results show that, under the source-centered workload of TDLight—where spatial queries are central and the data model is designed around per-source tables—the HEALPix-based per-source layout outperforms conventional single-table relational baselines by a factor of 12×–25× in cone-search latency, while also retaining the higher write throughput and lower storage footprint reported above.

[8] https://rubinobservatory.org/for-scientists/rubin-101/key-numbers

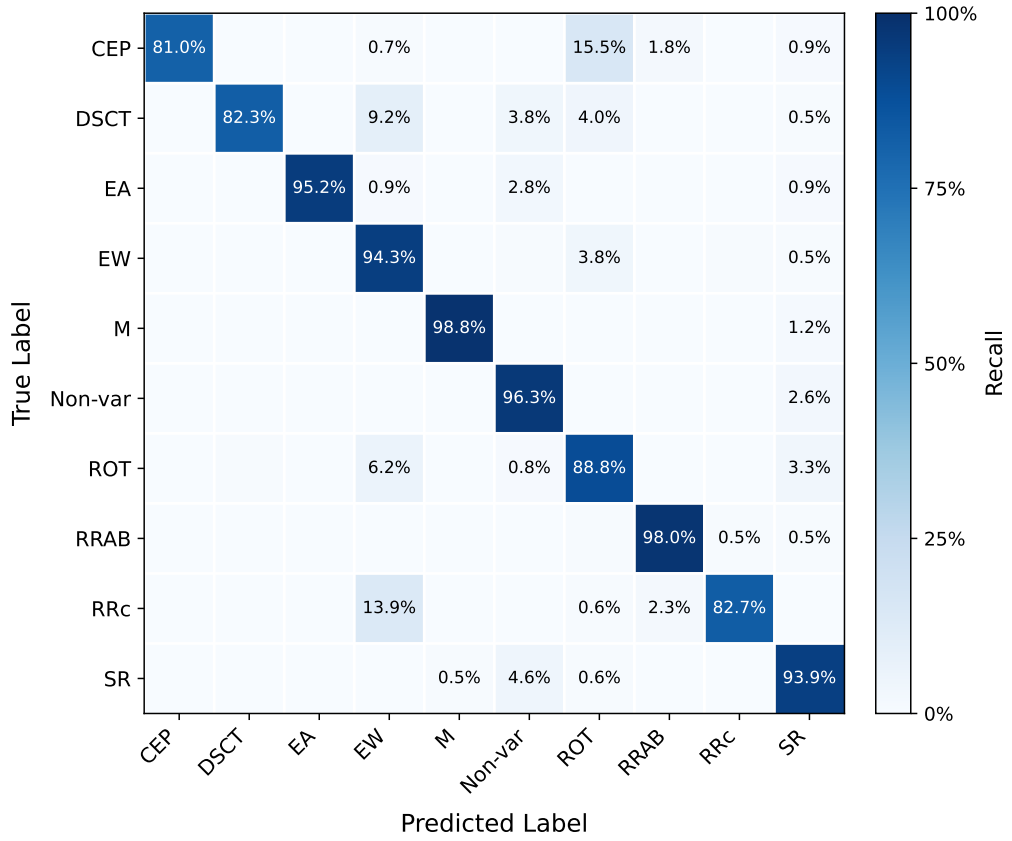

Figure 4. Normalized confusion matrix from 5-fold cross-validation on the LEAVES dataset (1.10 M sources, 10 classes). Diagonal entries denote per-class recall; off-diagonal entries indicate the dominant misclassification channels. The overall accuracy is 93.6%.

4.3. *Classification*

For classification, TDLIGHT uses the LEAVES dataset (Y. Fei et al. 2024), a publicly released multi-survey light-curve dataset constructed from ASAS-SN, ZTF, and *Gaia*, containing ∼1.10 million sources in 10 leaf classes. The original LEAVES balanced-random-forest (BRF) models achieved a weighted F1 score of 0.93 on the 10-class task but are too heavy for lightweight online deployment: their serialized files total 21.8 GB and require more than 22 GB of resident memory at runtime (Table 4), with a loading time of about 18 s. We therefore retain the same four-level hierarchical architecture but replace the BRF implementation with LightGBM (G. Ke et al. 2017), yielding a much more compact classifier for CPU-only deployment. The classification module is designed to be replaceable; the interface requirements are provided in Appendix A.

Model architecture. The classifier retains the same four-level decision tree used by LEAVES: (1) Variable vs. Non-variable, (2) Extrinsic vs. Intrinsic, (3) subclass routing (e.g. EB vs. ROT; RR vs. LPV vs. CEP vs. DSCT), and (4) fine-grained leaf classifiers (e.g. RRAB vs. RRc; EA vs. EW; Mira vs. SR). This yields seven individual LightGBM sub-models, each operating on 15 statistical features extracted by the FEETS package (J. B. Cabral et al. 2018), including the Lomb–Scargle period, Stetson K index, amplitude ratios, and variability measures. For production deployment, the trained models are further exported to ONNX format.

Training and accuracy. All seven sub-models were trained on the full LEAVES dataset and evaluated via stratified 5-fold cross-validation. The resulting ensemble achieves an overall accuracy of 93.6% and a weighted F1 score of 93.7% on the 10-class task (Figure 4). For comparison, the original LEAVES hierarchical RF baseline reports a weighted F1 score of 0.93 on the same 10-class setting. The difference in the overall weighted metric is modest. The original baseline shows stronger class-dependent variation, with particularly weak results for minority subclasses such as DSCT (F1 = 0.80) and low precision for CEP (0.62); our model exhibits a more even recall profile across classes.

Table 4. Inference comparison: LEAVES BRF vs. LightGBM. Full dataset (∼1.10 M sources), 64 CPU threads, Intel Xeon Gold 6530.

Metric	LEAVES BRF	LightGBM
Model size (disk)	21.8 GB	252 MB (87×)
Memory (RSS)	22.6 GB	<10 MB
Load time	18.2 s	4.5 s (4.0×)
Inference time	14.3 s	17.6 s
End-to-end latency	32.4 s	22.2 s (1.5×)

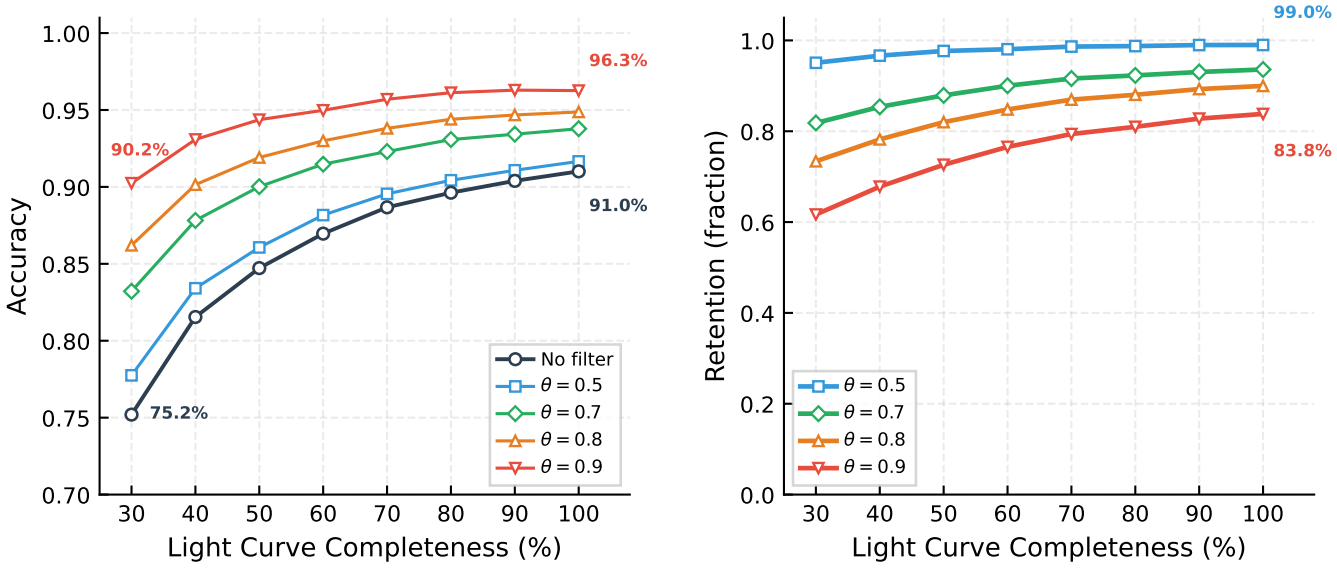

Figure 5. Confidence filtering vs. light-curve completeness. **(a)** Classification accuracy without filtering and with filtering on predictions satisfying $c \geq \theta$ for several θ. **(b)** Overall retention fraction (fraction of sources with $c \geq \theta$) for the same thresholds; stricter θ lowers retention.

In our model, per-class recall ranges from 81.0% (CEP) to 98.8% (M), with the dominant classes (EW, SR) both exceeding 93%.

Deployment efficiency. As shown in Table 4, the LightGBM ensemble reduces the total model footprint from 21.8 GB to 252 MB—an 87× reduction—while requiring less than 10 MB of runtime memory. Although the pure inference time is slightly higher than that of the original BRF model (17.6 s versus 14.3 s), the dominant bottleneck in practice is model loading rather than tree traversal, and the loading time drops from 18.2 s to 4.5 s. As a result, the end-to-end latency is reduced from 32.4 s to 22.2 s. Moreover, TDLIGHT is designed for incremental and on-demand classification during database updates, rather than for repeatedly loading and classifying the entire LEAVES corpus at once. Under this deployment pattern, the much smaller model footprint is more important than the modest increase in per-batch inference time. The resulting system runs on a single server without GPU support.

4.4. *Early Classification and Confidence Filtering*

Although the classifier was trained on complete light curves, real-time applications often rely on partial observations (e.g., D. Muthukrishna et al. 2019). We therefore performed a retrospective truncation experiment using the same 5-fold splits as in Section 4.3. In each fold, models were trained on complete training curves, while test curves were truncated to $f \in \{30\%, 40\%, \ldots, 100\%\}$ of full length. Features were re-extracted from the truncated test curves and

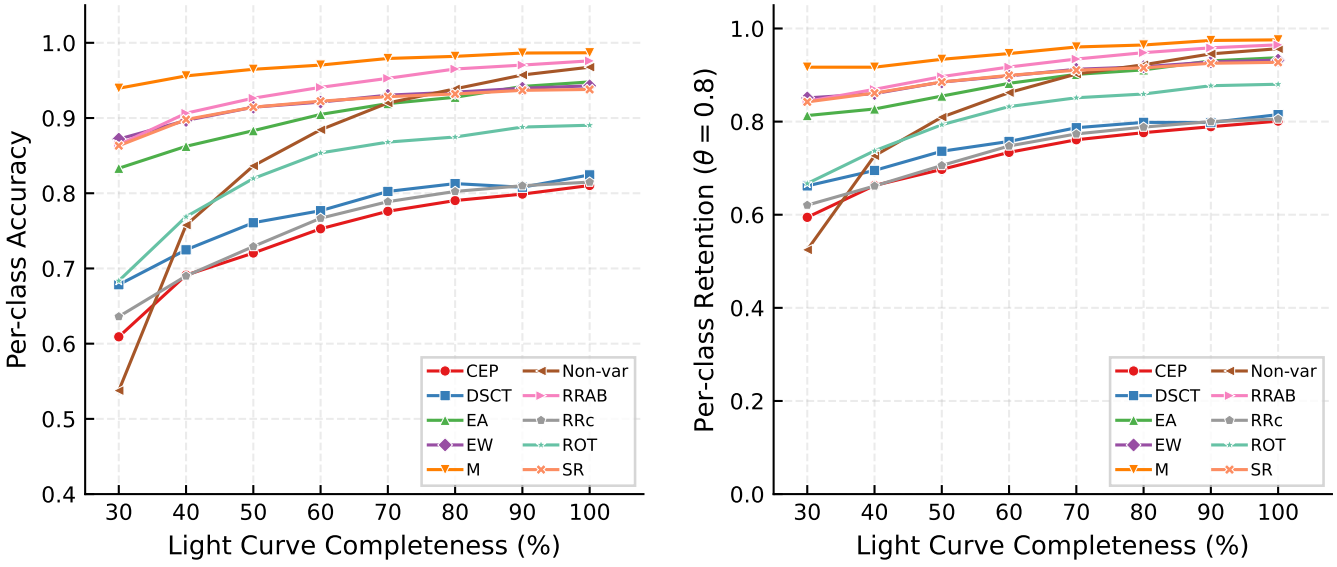

Figure 6. Per-class metrics at fixed write-back threshold $\theta = 0.8$. **(a)** Per-class classification accuracy vs. completeness. **(b)** Per-class retention (fraction of sources in each class with $c \geq \theta$).

evaluated with the fold-specific model. This is a controlled post hoc test of incomplete-curve performance, not a model explicitly trained for streaming classification.

For this truncation experiment, we sampled 1000 light curves from each class and applied seven truncation ratios, yielding about 70,000 truncated instances in total, which were evaluated under the standard 5-fold cross-validation inference setting. The overall accuracy increases monotonically with completeness, from 74.1% at f=30% to 97.4% at f=100%, reaching 85.0% at f=50%. High-amplitude variables such as Mira stars (M, 92.6% at f=30%) can already be identified with high accuracy at low completeness, whereas lower-amplitude classes such as Non-var, ROT, and RRc generally require $\geq 60\%$ completeness before their predictions become consistently accurate (Figure 6a).

Confidence filtering. To suppress unreliable predictions at early stages, we define a path-integrated confidence score for the hierarchical classifier. Because each source is routed through a sequence of sub-models, the confidence is computed as the product of the probabilities assigned to the selected branch at each decision layer. For example, a source classified as RRAB traverses four nodes (Variable → Intrinsic → RR → RRAB), and its confidence is

$$c = \prod_{l=1}^{L} p_l, \tag{1}$$

where p_l denotes the probability assigned to the selected class at layer l, and L is the length of the decision path. This score is used as a practical filtering criterion rather than as a calibrated posterior probability. Predictions with $c < \theta$ are withheld at the current stage and revisited after additional observations become available.

The confidence score in Eq. (1) is the default path-level confidence used in TDLIGHT. It is compared with a user-configurable threshold θ to control automatic database write-back. This path-integrated score is intentionally adopted as a conservative operating choice, because an apparently confident low-level classification may still be unreliable if an incorrect routing decision has already occurred at a higher level of the hierarchy. In this sense, the path product helps suppress premature write-back caused by upstream errors. At the same time, node-level conditional probabilities remain useful as diagnostic quantities and can be provided for branch-level inspection or user-defined operating policies. In practical use, the threshold θ can be adjusted according to the desired trade-off between retention and reliability.

Figure 5 shows the accuracy–retention trade-off under different θ: higher thresholds improve reliability at the cost of coverage.

Trigger policy. Classification is controlled by the confidence threshold θ and growth ratio g. Sources are selected if they are newly ingested and still unlabeled, or if their light curves have grown by more than g since the previous classification (default g=20% of the previous length). For each selected source, the system extracts 15 features, applies the hierarchical classifier, and computes the path-integrated confidence score c. Only predictions with $c \geq \theta$ (default

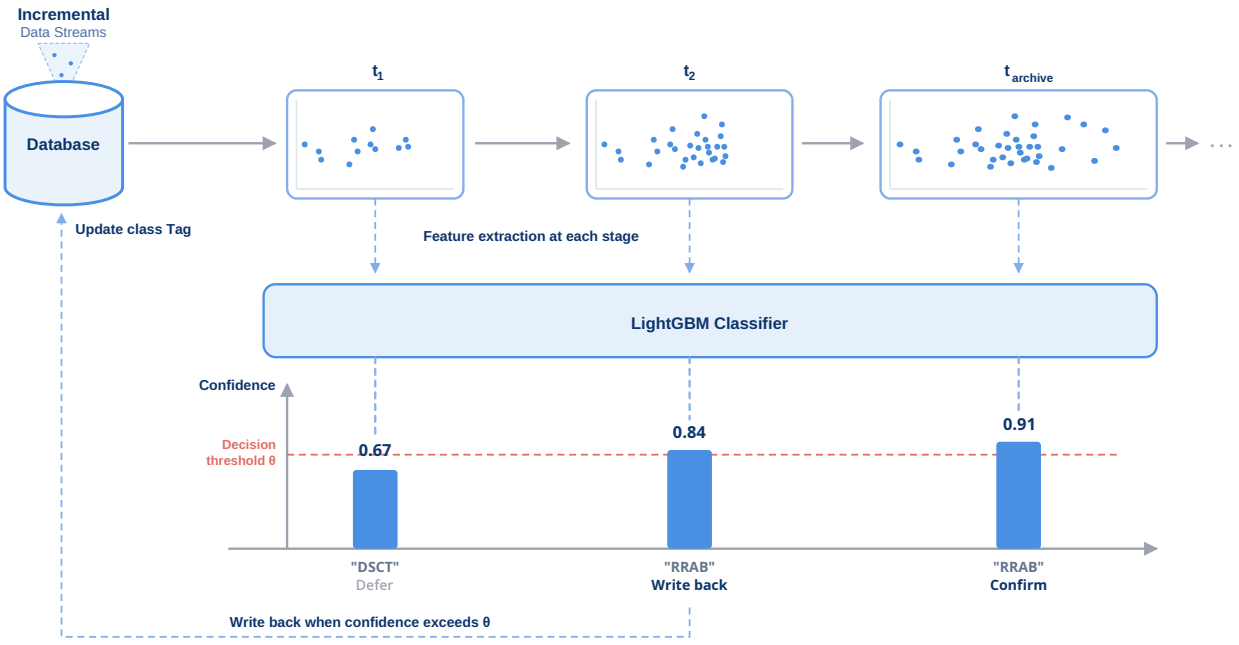

Figure 7. Incremental classification and confidence-based database write-back in TDLIGHT.

θ=0.8) are written back to the database; otherwise, classification is deferred until more observations accumulate. Classification can also be triggered manually through the web interface.

Design rationale. The default g=20% balances responsiveness against computational cost: smaller values lead to frequent reclassification with limited feature changes, whereas larger values delay potentially improved predictions. The default θ=0.8 is chosen as a practical operating point on the accuracy–retention curve (Figure 5a), improving reliability while retaining most sources. Because early-stage light curves are incomplete and may differ substantially from the complete curves used for training, confidence filtering helps prevent unreliable write-back and enables more reliable early classification.

4.5. *Catalog Disagreements and Candidate Label Issues in* LEAVES

Beyond incremental classification, TDLIGHT can also be used to identify potentially problematic catalog labels through high-confidence disagreements between model predictions and archived classifications. Among the 14,355 LEAVES sources successfully cross-matched with Simbad (M. Wenger et al. 2000) and assigned prediction confidence ≥ 0.95, 560 cases were identified in which the TDLIGHT prediction disagrees with the original LEAVES label but is supported by the Simbad classification.

To further assess these candidates, we cross-matched them against additional external variability catalogs, including VSX (C. L. Watson et al. 2006), ASAS-SN (C. S. Kochanek et al. 2017; T. Jayasinghe et al. 2018, 2019; C. T. Christy et al. 2023), and GCVS (N. N. Samus' et al. 2017). Of the 560 candidates, 402 (71.8%) are supported by at least two external catalogs in favor of the TDLIGHT prediction, while the remaining 158 (28.2%) show conflicting or insufficient evidence.

A particularly notable subset consists of 150 RRc→EW disagreements among the 402 externally supported candidates. All 150 sources exhibit an exact 2:1 ratio between the periods listed in LEAVES and the independently reported periods in VSX and ASAS-SN. This systematic offset is consistent with the analysis of T. Jayasinghe et al. (2020), who noted that some RRc variables in the ASAS-SN pipeline were assigned twice their true pulsation period and could consequently be confused with EW systems because of their relatively symmetric folded light-curve morphologies. Since the relevant LEAVES entries are derived from ASAS-SN, this pattern strongly suggests that the disagreement reflects an inherited period-doubling artifact rather than random model error, with the TDLIGHT prediction recovering the physically more plausible RRc interpretation.

Table 5. Source-by-source catalog of high-confidence disagreement candidates

Column #	Units	Label	Description
1	—	Index	Original input-catalog row index [row_index]
2	—	Name	Source name [source_name]
3	—	True	Original input-catalog class label [true_label]
4	—	Pred	TDLight predicted class label [pred_label]
5	—	Conf	TDLight path confidence [confidence]
6	d	Per	TDLight Lomb-Scargle period [period]
7	mag	Amp	Peak-to-peak amplitude [amplitude]
8	deg	RAdeg	Right ascension (J2000) [ra_deg]
9	deg	DEdeg	Declination (J2000) [dec_deg]
10	—	Simbad	SIMBAD identifier [simbad_id]
11	—	otype	SIMBAD object type [simbad_otype]
12	—	VSX	VSX name [vsx_name]
13	—	Type-VSX	VSX object type [vsx_type]
14	d	Per-VSX	VSX period [vsx_period]
15	—	ASASSN	ASAS-SN name [asassn_name]
16	—	Type-ASASSN	ASAS-SN object type [asassn_type]
17	d	Per-ASASSN	ASAS-SN period [asassn_period]
18	mag	Amp-ASASSN	ASAS-SN amplitude [asassn_amp]
19	—	GCVS	GCVS name [gcvs_name]
20	—	Type-GCVS	GCVS object type [gcvs_type]
21	d	Per-GCVS	GCVS period [gcvs_period]
22	—	Pred-broad	Predicted broad class [pred_broad]
23	—	True-broad	Broad class of original label [true_broad]
24	—	VSX-broad	Broad class from VSX type [vsx_broad]
25	—	ASASSN-broad	Broad class from ASAS-SN type [asassn_broad]
26	—	Simbad-broad	Broad class from SIMBAD type [simbad_broad]
27	—	Ncpred	Catalog count supporting predicted broad class [n_confirm_pred]
28	—	Nctrue	Catalog count supporting original broad class [n_confirm_true]
29	—	Final	Final cross-catalog assessment [final_verdict]

Note—The labels in square brackets in this descriptive table correspond to the labels used for the data file available on Zenodo.

Note—Broad-class labels merge detailed types for cross-catalog comparison (e.g., EA/EW → EB; RRAB/RRc → RR; SR/M → LPV). MODEL_LIKELY_CORRECT means Ncpred > Nctrue; otherwise CONFLICTING.

Note—Per is the TDLight Lomb-Scargle feature value and may differ from external-catalog periods, especially for some long-period sources.

Note—Table 5 is published in its entirety in the electronic edition of the *Astrophysical Journal*. A portion is shown here for guidance regarding its form and content.

These results suggest that a substantial fraction of the observed disagreements are associated with systematic labeling issues in upstream survey products, rather than arbitrary classifier failures. TDLight therefore provides value not only as an incremental classification framework, but also as a practical tool for catalog auditing by highlighting candidates for label verification, period re-analysis, and follow-up inspection. The source-by-source catalog of the 560 high-confidence disagreement candidates identified in this work is available on Zenodo: doi:10.5281/zenodo.19736413 and summarized in Table 5.

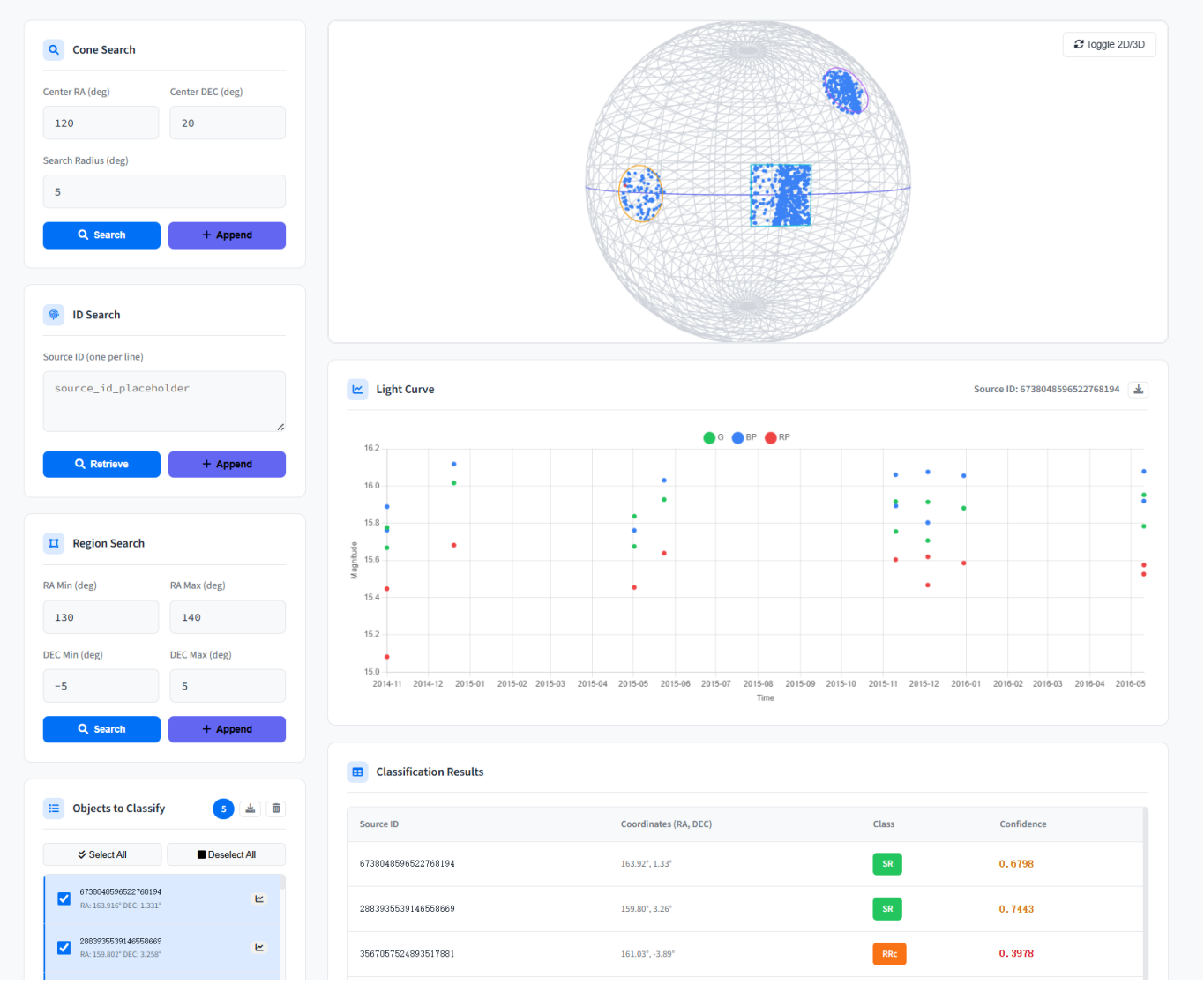

Figure 8. The TDLIGHT web interface. The left panel shows spatial querying; the right panel shows light curves and classification metadata.

5. DEPLOYMENT AND INTERFACE

5.1. *Open-Source Release and One-Click Installation*

TDLIGHT is released as an open-source project on GitHub. An automated installation script (`install.sh`) sets up TDENGINE, compiles the C++ ingestion and query modules, configures HEALPix, and deploys the web server and classification engine. The pre-trained LightGBM model is hosted on Hugging Face and can be integrated directly after installation.

5.2. *Web Interface*

TDLIGHT provides a browser-based frontend for cone and rectangular spatial queries, source lookup, light-curve inspection, and on-demand reclassification. A built-in data-import panel allows uploading light-curve archives and epoch catalogs through the GUI, with real-time progress monitoring via Server-Sent Events (Figure 8).

5.3. *Interactive Label Correction and Incremental Model Update*

In addition to automatic trigger-based classification, TDLIGHT also supports user-guided model refinement through the web interface. Users may upload source-level light-curve files in CSV or ZIP format, optionally providing corrected

labels through a `class`, `label`, or `type` field. The system then extracts the same 15 statistical features used by the online classifier and stores them as incremental training samples for the corresponding branches of the hierarchical model.

Rather than retraining the full classifier from scratch, TDLight updates only the affected LightGBM sub-models in the four-level hierarchy. This design preserves the lightweight deployment target of the system while enabling rapid incorporation of user feedback. The updated models can then be re-exported to ONNX format and reused by the online classification pipeline.

To improve robustness in practical use, the incremental update procedure includes several safety mechanisms, including automatic model backup before training, rollback on failure, and deferred ONNX export in the background. This design extends confidence-based database write-back to a human-in-the-loop workflow, where expert corrections can be incorporated into the deployed classifier without interrupting the surrounding storage and query services.

6. CONCLUSIONS AND OUTLOOK

We have presented TDLight, a database-centered framework that integrates incremental light-curve ingestion, repeated classification, and confidence-based database write-back within a single workflow. On a Gaia-derived workload, the system achieves archival ingestion throughput of $\sim 1.5 \times 10^6$ rows s^{-1} and supports sub-second spatial queries on a single server without GPU support. On the LEAVES benchmark, the hierarchical LightGBM classifier reaches an overall accuracy of 93.6%. Because classification is re-triggered as light curves grow, the confidence filter allows reliable predictions to be accepted before the full observing campaign is complete. Beyond online classification, TDLight also identifies 560 high-confidence catalog-disagreement candidates, indicating its potential utility for catalog verification and follow-up analysis.

Two directions for future work are particularly promising. First, TDengine supports distributed deployment across multiple nodes, and extending the current single-server implementation to a distributed setting would enable the same workflow to scale to much larger time-domain streams. Second, the classification module is designed to be replaceable: more expressive models for irregular and incrementally growing light curves, including Transformer-based approaches, could be incorporated without changing the surrounding ingestion and storage pipeline. These extensions would further improve the applicability of TDLight to large-scale time-domain surveys.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (NSFC; 12273025, 12133010, 62402333). Data resources are supported by the China National Astronomical Data Center (NADC) and the Chinese Virtual Observatory (China-VO). We also acknowledge TDengine as the open-source time-series database that underpins this work.

We thank the authors of the LEAVES project (Y. Fei et al. 2024) for publicly releasing their dataset and hierarchical classification resources, which enabled the evaluation and comparison presented in this work. We thank Las Cumbres Observatory and its staff for their continued support of ASAS-SN. ASAS-SN is funded in part by the Gordon and Betty Moore Foundation through grants GBMF5490 and GBMF10501 to the Ohio State University, and in part by the Alfred P. Sloan Foundation through grant G-2021-14192.

This work has made use of data from the European Space Agency (ESA) mission Gaia (https://www.cosmos.esa.int/gaia), processed by the Gaia Data Processing and Analysis Consortium (DPAC, https://www.cosmos.esa.int/web/gaia/dpac/consortium). Funding for the DPAC has been provided by national institutions, in particular the institutions participating in the Gaia Multilateral Agreement. This work also makes use of observations obtained with the Samuel Oschin 48-inch Telescope and the 60-inch Telescope at the Palomar Observatory as part of the Zwicky Transient Facility project. ZTF is supported by the National Science Foundation under grant Nos. AST-1440341 and AST-2034437, and by a collaboration including Caltech, IPAC, the Oskar Klein Center at Stockholm University, the University of Maryland, the University of California, Berkeley, the University of Wisconsin at Milwaukee, the University of Warwick, Ruhr University, Cornell University, Northwestern University, and Drexel University. Operations are conducted by COO, IPAC, and UW.

APPENDIX

A. CLASSIFICATION MODULE INTERFACE

The classification module in TDLight is decoupled from the database, ingestion trigger, and web interface through explicit input and output interfaces. In the released implementation (GitHub: https://github.com/bestdo77/TD-light), `class/classify_pipeline.py` and `class/auto_classify.py` handle database I/O and write-back, `class/hierarchical_predictor.py` implements the model adapter, and `web/app.js` reads the classification results through REST endpoints defined in `web/web_api.cpp`. A replacement classifier can therefore be integrated by preserving the model-side interfaces summarized in Table 6.

Table 6. Model-side interfaces around the TDLight classification module.

Interface	Type	API and content
Input	**API:**	`classify_pipeline.py, auto_classify.py: fetch_lightcurve()`
	Content:	retrieves `ts`, `mag`, and `mag_error` as `(t, mag, err)`
	API:	`classify_pipeline.py: extract_features()`
	Content:	converts light curves into the 15-dimensional feature vector used by the model
Output	**API:**	`hierarchical_predictor.py: predict()`
	Content:	returns the predicted class label and confidence score
	API:	`web_api.cpp: /api/classify_results`
	Content:	exposes result records with `source_id`, `prediction`, and `confidence`
	API:	`app.js: displayResults()`
	Content:	displays classification results and optional metadata fields

Under this interface, a replacement module may use a different light-curve encoder or classifier, provided that it accepts the source-level light-curve or feature inputs supplied by the pipeline and returns the result fields consumed by the frontend and threshold-controlled database write-back logic. In the current implementation, database updates are performed by `classify_pipeline.py: update_tdengine_class()` and `auto_classify.py: update_class_in_db()`, which write the predicted label back to TDengine only when the returned score exceeds θ.

Software: TDengine v3.4, HEALPix (K. M. Górski et al. 2005), feets v0.4 (J. B. Cabral et al. 2018), LightGBM (G. Ke et al. 2017), ONNX Runtime v1.17, Python v3.9, scikit-learn v1.1 (F. Pedregosa et al. 2011)

REFERENCES

Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017, The Astrophysical Journal Letters, 848, L12, doi: 10.3847/2041-8213/aa91c9

Alcock, C., Allsman, R. A., Alves, D. R., et al. 2000, The Astrophysical Journal, 542, 281, doi: 10.1086/309512

Bellm, E. C., Kulkarni, S. R., Graham, M. J., et al. 2019, Publications of the Astronomical Society of the Pacific, 131, 018002, doi: 10.1088/1538-3873/aaecbe

Budavári, T., & Szalay, A. S. 2008, The Astrophysical Journal, 679, 301, doi: 10.1086/587156

Cabral, J. B., Sánchez, B., Ramos, F., et al. 2018, Astronomy and Computing, 25, 213, doi: 10.1016/j.ascom.2018.09.005

Christy, C. T., Jayasinghe, T., Stanek, K. Z., et al. 2023, Monthly Notices of the Royal Astronomical Society, 519, 5271, doi: 10.1093/mnras/stac3801

Clementini, G., Ripepi, V., Molinaro, R., et al. 2019, Astronomy & Astrophysics, 622, A60, doi: 10.1051/0004-6361/201833374

Collaboration, G. 2018, Gaia DR2, European Space Agency, doi: 10.5270/esa-ycsawu7

Collaboration, G. 2022, Gaia DR3, European Space Agency, doi: 10.5270/esa-qa4lep3

D'Isanto, A., Cavuoti, S., Brescia, M., et al. 2016, Monthly Notices of the Royal Astronomical Society, 457, 3119, doi: 10.1093/mnras/stw157

Evans, D. W., Riello, M., De Angeli, F., et al. 2018, Astronomy & Astrophysics, 616, A4, doi: 10.1051/0004-6361/201832756

Fei, Y., Yu, C., Li, K., et al. 2024, The Astrophysical Journal Supplement Series, 275, 10, doi: 10.3847/1538-4365/ad785b

Fluke, C. J., & Jacobs, C. 2020, Wiley Interdiscip. Rev.: Data Mining and Knowl. Discov., 10, e1349, doi: 10.1002/widm.1349

Förster, F., Cabrera-Vives, G., Castillo-Navarrete, E., et al. 2021, The Astronomical Journal, 161, 242, doi: 10.3847/1538-3881/abe9bc

Gaia Collaboration, Brown, A. G. A., Vallenari, A., et al. 2018, Astronomy & Astrophysics, 616, A1, doi: 10.1051/0004-6361/201833051

Gaia Collaboration, Vallenari, A., Brown, A. G. A., et al. 2023, Astronomy & Astrophysics, 674, A1, doi: 10.1051/0004-6361/202243940

Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, The Astrophysical Journal, 622, 759, doi: 10.1086/427976

Holl, B., Audard, M., Nienartowicz, K., et al. 2018, Astronomy & Astrophysics, 618, A30, doi: 10.1051/0004-6361/201832892

Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, The Astrophysical Journal, 873, 111, doi: 10.3847/1538-4357/ab042c

Jayasinghe, T., Kochanek, C. S., Stanek, K. Z., et al. 2018, Monthly Notices of the Royal Astronomical Society, 477, 3145, doi: 10.1093/mnras/sty838

Jayasinghe, T., Stanek, K. Z., Kochanek, C. S., et al. 2019, Monthly Notices of the Royal Astronomical Society, 486, 1907, doi: 10.1093/mnras/stz844

Jayasinghe, T., Stanek, K. Z., Kochanek, C. S., et al. 2020, Monthly Notices of the Royal Astronomical Society, 493, 4045, doi: 10.1093/mnras/staa518

Ke, G., Meng, Q., Finley, T., et al. 2017, in Advances in Neural Information Processing Systems, Vol. 30, 3146–3154

Kim, D.-W., Protopapas, P., Bailer-Jones, C. A. L., et al. 2014, Astronomy & Astrophysics, 566, A43, doi: 10.1051/0004-6361/201323252

Kochanek, C. S., Shappee, B. J., Stanek, K. Z., et al. 2017, Publications of the Astronomical Society of the Pacific, 129, 104502, doi: 10.1088/1538-3873/aa80d9

Lomb, N. R. 1976, Astrophysics and Space Science, 39, 447, doi: 10.1007/BF00648343

Masci, F. J., Laher, R. R., Rusholme, B., et al. 2019, Publications of the Astronomical Society of the Pacific, 131, 018003, doi: 10.1088/1538-3873/aae8ac

Muthukrishna, D., Narayan, G., Mandel, K. S., Biswas, R., & Hlǒžek, R. 2019, Publications of the Astronomical Society of the Pacific, 131, 118002, doi: 10.1088/1538-3873/ab1609

Patterson, M. T., Bellm, E. C., Rusholme, B., et al. 2019, Publications of the Astronomical Society of the Pacific, 131, 018001, doi: 10.1088/1538-3873/aae904

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, Journal of Machine Learning Research, 12, 2825. http://jmlr.org/papers/v12/pedregosa11a.html

Perlmutter, S., Aldering, G., Goldhaber, G., et al. 1999, The Astrophysical Journal, 517, 565, doi: 10.1086/307221

Richards, J. W., Starr, D. L., Butler, N. R., et al. 2011, The Astrophysical Journal, 733, 10, doi: 10.1088/0004-637X/733/1/10

Riess, A. G., Filippenko, A. V., Challis, P., et al. 1998, The Astronomical Journal, 116, 1009, doi: 10.1086/300499

Riess, A. G., Macri, L. M., Hoffmann, S. L., et al. 2016, The Astrophysical Journal, 826, 56, doi: 10.3847/0004-637X/826/1/56

Samus', N. N., Kazarovets, E. V., Durlevich, O. V., Kireeva, N. N., & Pastukhova, E. N. 2017, Astronomy Reports, 61, 80, doi: 10.1134/S1063772917010085

Scargle, J. D. 1982, The Astrophysical Journal, 263, 835, doi: 10.1086/160554

Udalski, A., Szymański, M. K., & Szymański, G. 2015, Acta Astronomica, 65, 1

Watson, C. L., Henden, A. A., & Price, A. 2006, in Society for Astronomical Sciences Annual Symposium, Vol. 25, 47

Wenger, M., Ochsenbein, F., Egret, D., et al. 2000, Astronomy and Astrophysics Supplement Series, 143, 9, doi: 10.1051/aas:2000332

Yu, C. 2024, LEAVES, NADC, doi: 10.12149/100962

Yu, C., Li, K., Tang, S., et al. 2020, Monthly Notices of the Royal Astronomical Society, 496, 629, doi: 10.1093/mnras/staa1413

Yu, C., Li, K., Zhang, Y., et al. 2021, Wiley Interdiscip. Rev.: Data Mining and Knowl. Discov., 11, e1425, doi: 10.1002/widm.1425

Zhang, Z., Xu, Y., Cui, C., & Fan, D. 2024, Astronomy and Computing, 48, 100845, doi: 10.1016/j.ascom.2024.100845

ZTF Team. 2025, ZTF Lightcurves, IPAC, doi: 10.26131/IRSA598

Zuo, X., Tao, Y., Huang, Y., et al. 2026, The Astronomical Journal, 171, 10, doi: 10.3847/1538-3881/ae1467