DcDM: A Pre-training Data Evaluation Framework for Proactive Drift Prevention in Machine Learning

Okjoo Choi[1], Wonsun Shin[2]
Dept. of AI·Software Engineering, PAI CHAI University, Daejeon, Korea, Republic of Korea[1]
Dept. of Smart ICT Convergence, PAI CHAI University, Daejeon, Korea, Republic of Korea[2]

***Abstract*—The performance of machine learning (ML) systems often deteriorates over time owing to data drift, which is typically caused by changes in data quality or distribution. Such degradation in deployment environments can result in inaccurate predictions and reduced system reliability. Conventional drift detection approaches have largely focused on retraining ML models after performance degradation has occurred. However, because the root causes of drift often originate from the data itself, a data-centric approach is needed to address the problem at its source. This study proposes a systematic, data-centric drift management (DcDM) framework that integrates domain-specific rule validation, data quality assessment, and statistical drift analysis before model training. By first verifying semantic constraints and then assessing data quality and distributional stability, DcDM enables early identification of potential drift hazards and prevents low-quality or semantically invalid data from entering the training pipeline. We evaluate the proposed framework on three datasets across different modalities: Electricity Load Diagrams, CIFAR-10, and VisDA-2017. The experimental results show consistent improvements of approximately 6% to 12.5% in both accuracy and F1-score. Additional ablation studies, baseline comparisons, and statistical significance tests further demonstrate the robustness and effectiveness of the proposed approach.**

***Keywords*—Data drift; data evaluation; data quality; domain rule; drift prevention and mitigation; drift management**

I. Introduction

Machine learning (ML) systems have become indispensable across a wide range of applications, including image classification, speech recognition, financial forecasting, and medical diagnostics. As ML adoption expands across diverse industrial domains, ensuring the accuracy and long-term reliability of deployed models has become increasingly critical [1]. A primary challenge faced by these systems is data drift, a phenomenon in which the statistical properties of input data change over time, potentially degrading model performance [2]. For example, a model trained to predict a company's stock price using historically stable market data may initially perform well; however, as market volatility increases, the shifting data distribution can render inaccurate predictions [3]. Such drift-induced degradation is especially problematic in real-world environments where continuous reliability is paramount [4].

Traditional responses to data drift typically adopt model-centric strategies, such as retraining or fine-tuning models once performance decline is observed [5]. While these methods can offer temporary recovery, they are frequently time-consuming and computationally expensive, particularly in dynamic environments where operational data changes rapidly [6]. Moreover, such approaches fail to address the root cause of drift, which frequently originates from the data itself rather than from the model [7]. In contrast, data-centric drift management emphasizes maintaining data quality and semantic integrity throughout the ML lifecycle [8][9]. This paradigm focuses on the systematic evaluation of data properties to proactively detect and mitigate drift before it impacts model performance [10].

In this study, we propose DcDM, a unified pre-training framework for proactive drift prevention. The proposed framework is not the introduction of isolated techniques for drift management, but the integration of domain rule validation, data quality assessment, and statistical drift analysis into a single upstream pipeline. This design differs from conventional MLOps and drift-monitoring tools, which are primarily reactive and operate after deployment. DcDM is intended to detect drift hazards before model training and to filter or refine problematic data before they affect downstream learning. This study makes four contributions. First, it introduces a unified data-centric framework that combines domain rule validation, quality assessment, and drift analysis in a single workflow. Second, it operationalizes this framework as an end-to-end pipeline for pre-training data evaluation. Third, it demonstrates the framework on heterogeneous datasets spanning tabular time-series and image domains. Fourth, it performs controlled ablation and baseline comparisons to quantify the contribution of each pipeline component and to distinguish DcDM from simple data cleansing.

To demonstrate the effectiveness and generalizability of the proposed framework, we conducted experiments on three benchmark datasets from different domains: electricity load diagrams (time-series), CIFAR-10 (image classification), and VisDA-2017 (domain adaptation). The experimental results reveal significant improvements, ranging from approximately 6% to 12.5% in both accuracy and F1-score, across all datasets. These findings highlight the utility of proactive data evaluation in preserving the predictive performance and mitigating the risk of drift-induced degradation in ML systems.

II. Related Work

Managing the performance degradation caused by data drift has been extensively studied in recent years. The existing research can be broadly categorized into three areas: 1) drift types, 2) drift detection methods, and 3) drift management frameworks, including data-centric approaches.

A. Drift Types in ML

Drift in ML systems refers to the change in data properties over time, resulting in a potential mismatch between training and operational environments. There are two primary types of drift: data drift (covariate shift) and concept drift. Data drift involves a change in the distribution of input features *P(x)*, whereas concept drift reflects changes in the conditional distribution *P(y|x)* over time.

1) Data drift: Data drift is defined as a change in the distribution of input data relative to the distribution present during model training.

$$P_{tran}(x) \neq P_{operational}(x)$$

For example, noise due to age-related deterioration of camera lenses or out-of-focus images constitutes data drift, which affects classification performance.

2) Concept drift: Concept drift occurs when the relationship between input and target variables changes relative to the relationship learned during model training.

$$P_{tran}(y|x) \neq P_{operational}(y|x)$$

For example, a machine learning model trained to detect spam emails based on content may fail when spam patterns evolve significantly. Concept drift can be further categorized into four main categories: sudden, gradual, incremental, and recurring drift [11].

B. Drift Detection Methods

Identify applicable sponsor/s here. If no sponsors, delete this text box (*sponsors*).

Early drift detection studies mainly focused on statistical monitoring approaches that compare operational data distributions with baseline training distributions.

1) Statistics methods: Statistical test methods measure distribution changes between two datasets (training vs. current). These include:

a) Population Stability Index (PSI): PSI assesses whether training and operational data come from the same distribution [12]. Lower PSI values indicate more similar, stable distributions.

b) Kullback-Leibler(KL) Divergence: KL divergence quantitatively measures the extent to which the distribution of operation data differs from training data [13]. An increase beyond a threshold signals drift.

c) Jensen-Shannon(JS) Divergence: JS divergence is a symmetrized and smoothed version of KL divergence, bounded between 0 and 1, providing a normalized similarity measure [14].

d) Kolmogorov-Smirnov(KS) Test: The KS test compares cumulative distribution functions (CDFs) of two datasets. A significant p-value (< 0.05) indicates distributional differences signalling potential data drift [15].

2) Streaming methods: Streaming methods monitor statistical measures such as mean, variance, skewness, and kurtosis over time. These include:

a) Drift Detection Method (DDM): [16]

b) Early Drift Detection Method (EDDM) [17]

c) Exponentially Weighted Moving Average (EWMA) [18].

3) Time window-based approaches: These methods typically use a fixed reference window, summarize historical information, and a sliding window that summarizes up-to-date information. Dividing data into time windows and comparing statistics or distributions across these windows can help detect gradual changes in the data over time. A significant difference in the distributions of these windows indicates that a drift has occurred. These methods include adaptive windowing (ADWIN) [19] and hopping drift detection (HDDMA and HDDMW) tests [20].

4) Representation-space drift detection:

a) MMD(Maximum Mean Discrepancy): MMD is a kernel-based statistical test that measures the distance between the mean embeddings of two distributions in a reproducing kernel Hilbert space. It is particularly suited to high-dimensional data such as images and text, and has been applied to drift detection in deep learning contexts [21].

b) FID(Fréchet Inception Distance): FID measures the distance between feature distributions of real and generated (or reference vs. operational) image datasets by comparing the mean and covariance of Inception-v3 feature vectors. It is widely used in image-based drift detection [22].

These drift detection methods are widely used for structured data, whereas unstructured data often requires perceptual or embedding-based approaches. Table I presents the classification of drift detectors by data type.

TABLE I. DRIFT DETECTION METHOD BY DATA TYPES

Data Type	***Method***	***Reason***
Structured (Tabular)	PSI, KS-Test, KL/JS, ADWIN, DDM, EDDM	Well-suited for numerical and categorical data with a defined schema
Image	FID, LPIPS, MMD, KL/JS, HDDDM, Classifier-based	Support perceptual similarity and high-dimensional shifts
Audio	KL/JS, MMD, Page-Hinkley, Classifier-based	Use spectrograms or embeddings for detection
Text	KL/JS (embeddings), MMD, Classifier-based	Apply embedding models like BERT for drift detection
Streaming	ADWIN, DDM, EDDM, Page-Hinkley	Real-time monitoring of concept drift

C. AutoML-Based Method

Celik proposed strategies to address conceptual drift in AutoML environments using Bayesian optimization and genetic programming [23]. AML4S is a framework that automates ML model selection and hyperparameter tuning to adapt the model in real time when performance drops below a threshold (approximately 7%) [24]. An AutoML-DC model utilizing

meta-learning and automated ensemble learning was proposed to correct for incremental conceptual drift in sensor data [25]. These studies remain focused on model retraining and address issues within limited scopes, such as IoT environments or sensor data.

D. Data-Centric AI and MLOps

The data-centric AI paradigm, advocated by Ng [26], shifts focus from model architecture optimization to systematic improvement of data quality, labeling, and validation. In this view, fixing the data yields more reliable gains than iterating on the model alone. MLOps frameworks such as Evidently AI [27] and Alibi-Detect [28] provide open-source tooling for monitoring data distributions and detecting drift in production pipelines. However, these tools primarily operate reactively—alerting practitioners after drift symptoms emerge—and do not integrate domain-specific semantic rule validation into a unified pre-training pipeline. DcDM complements and extends these tools by adding upstream semantic validation and quality gating before model training.

III. Research Background

Although considerable progress has been made in drift detection and MLOps automation, existing solutions often address only isolated stages of the machine learning lifecycle. Most prior methods focus on model retraining or statistical detection after drift symptoms have already manifested in deployed systems. Consequently, these approaches tend to be reactive rather than proactive, which can lead to delayed corrective actions and inefficient model maintenance cycles.

In addition, conventional drift-handling frameworks seldom incorporate domain-specific constraints or data quality metrics at the pre-modeling stage. This lack of early validation often results in the propagation of low-quality or semantically invalid data into the training pipeline, thereby compromising the robustness and generalization ability of the model.

To overcome these limitations, we introduce a data-centric drift management (DcDM) framework that proactively monitors and evaluates data throughout the ML lifecycle. The DcDM framework aims to:

- Prevent drift hazards by validating data against business and domain-specific rules (semantic validation).
- Quantify drift risks by assessing quality metrics and statistical indicators before model training.
- Enable early interventions, such as data cleansing or discarding, before drift-induced performance degradation occurs.

This approach is grounded in the data-centric AI paradigm, which prioritizes data quality and integrity over model-centric optimization alone. Our contribution is the integration of rule-based evaluation, quality assessment, and statistical drift detection into a unified workflow, enabling ML systems to be more resilient to dynamic, real-world data environments. Table II lists the key terms defined in this study.

TABLE II. Key Terms

Term	Definition
Drift Hazard	A potential anomaly in the input data that could lead to degraded ML performance, identifiable before model training or inference.
Drift Risk	The probability that a drift hazard will actually lead to performance degradation during or after prediction.
Drift Prevention	A proactive strategy to detect and assess incoming data for potential hazards. If anomalies are detected—whether due to shifts in data distribution or quality violations—the system responds by collecting new data or retraining the model to realign with the original objectives.
Drift Mitigation	The process of excluding data that fails domain rule validation or quality checks from model training and inference, thereby minimizing the risk of drift.

Fig. 1 illustrates potential risks and hazards associated with drift across the ML lifecycle.

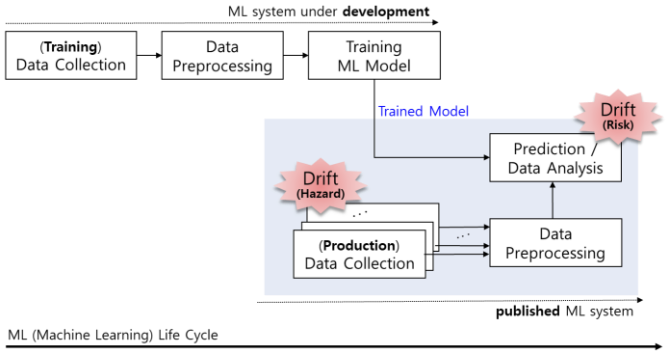

Fig. 1. Drift hazard and risk.

In contrast to previous studies that focus on detecting and responding to drift after performance deterioration, our approach emphasizes early-stage intervention. By systematically assessing data before it influences the model, our framework enables both drift prevention and effective mitigation.

IV. Data-Centric Drift Management

This study proposes a data-centric drift management (DcDM) framework that integrates data validation and drift detection into the ML pipeline. The proposed framework is structured into three steps—data requirement, data evaluation, and model performance analysis—to ensure that only clean, reliable data proceed to model training and inference. The data evaluation process comprises three key components: domain rule validation, data quality assessment, and drift analysis. This stepwise data evaluation aims at ensuring semantic validity by enforcing domain-specific constraints and confirming technical reliability using established quality metrics before drift detection. The DcDM framework is shown in Fig. 2, effectively prevents drift hazards and identifies and mitigates the performance degradation caused by drift.

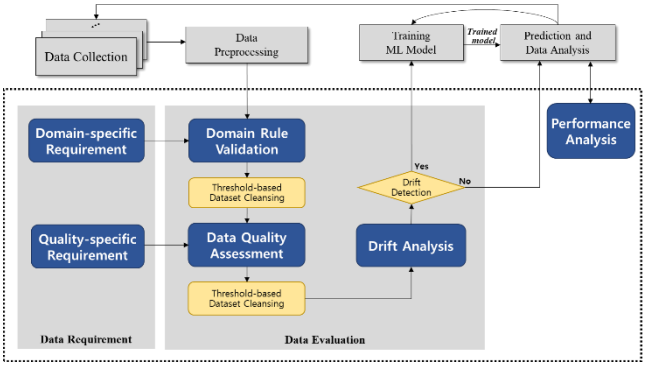

Fig. 2. Data-centric Drift Management (DcDM) framework.

A. Data Requirement

The first step establishes explicit data requirements that incorporate both business-specific rules and quality-based constraints. These requirements serve as a foundation for subsequent validation, monitoring, and drift mitigation activities.

1) Domain-specific requirement: Domain-specific requirements define the business-driven constraints that the dataset must satisfy. These are formalized as domain rules DR_i, which define valid value ranges [α_low_i, α_high_i], or logical inter-attribute relationships. Rules are typically derived from domain expertise, empirical guidelines, or regulatory standards and serve as primary checkpoints for semantic validity. Beyond univariate constraints, inter-attribute rules (e.g., the number of confirmed infection cases must not exceed the number of individuals tested) preserve relational integrity.

2) Quality-specific requirement: High-quality data is essential for preventing data drift. This study adopts goal-based data quality metrics (DQM) tailored to each application domain. For structured data as Table III, four metrics are quantified: accuracy, completeness, consistency, and redundancy.

TABLE III. STRUCTURED DATA: QUALITY METRICS AND THRESHOLDS

Metrics	Threshold	Violation Rate and Judgment		
		Keep (≤)	***Cleansing*** (≤)	***Discard*** (>)
Accuracy	0.95	5%	15%	15%
Completeness	0.95	5%	20%	20%
Consistency	0.95	10%	30%	30%
Redundancy	0.95	5%	20%	20%

For unstructured data such as images, audio, and text, domain-adapted metrics assess semantic integrity, content usability, and statistical coherence, as Table IV. Note that for low-resolution tasks (e.g., CIFAR-10 at 32×32), resolution thresholds and perceptual metrics are adjusted accordingly to reflect realistic quality expectations for the target data modality.

B. Data Evaluation

In this step, data are assessed to ensure compliance with the goal-oriented requirements defined by domain-specific rules and quality criteria. Subsequently, drift detection is performed on data meeting these criteria. The key idea is to pre-evaluate the integrity and usability of the operational dataset (D_op) using the clean reference dataset (D_ref) as a benchmark before training or inference.

TABLE IV. UNSTRUCTURED DATA: QUALITY METRICS AND THRESHOLDS

Data Type	Metric	Description	Threshold / Evaluation
Text	Readability Score	Assesses sentence clarity and grammar	≥ 60 readability score
	Language Consistency	Consistent language use across documents	≥ 95% consistent tokens
Image	Resolution Quality	Pixel density meets the application requirement	≥ 224x224 pixels
	Clarity / Blur Score	Measured using SSIM or Laplacian variance	SSIM ≥ 0.90 or LapVar ≥ 100
	Redundancy Detection	Duplicate image detection	≤ 5% duplicates
Audio	Signal-to-Noise Ratio	Quantifies audio clarity vs noise	SNR ≥ 20 dB
	Duration Validity	Consistent duration across samples	[1s – 10s]
	Transcription Accuracy	ASR transcription confidence	≥ 95% WER-based confidence

Algorithm 1 formalizes the three-step evaluation process. Thresholds α, β, and γ are selected empirically using a validation holdout set: the optimal values (α=10%, β=10%, γ=0.05) were identified through a grid search over the Electricity dataset and subsequently confirmed via the sensitivity analysis reported in Section V-E.

Algorithm 1: Data Evaluation

Input
  D_ref: reference dataset (baseline)
  D_op: operational dataset (current or incoming)
  DR: set of domain rules {DR_1, DR_2, ..., DR_n}
  DQM: set of data quality metrics {DQM_1, DQM_2, ..., DQM_m}
  α: rule violation threshold
  α_high: rule violation discard threshold
  β: data quality threshold
  β_high: data quality discard threshold
  γ: drift detection threshold
Output:
  Decision on D_op: {Accept, Cleansing, Discard}
  Drift Report: {Type, Magnitude, Location}

Step I: Domain Rule Validation
  for each domain rule DR_i in DR do
    ViolationRate[DR_i] ← EvaluateViolation(D_op, DR_i)
  end for
  v_total ← Average(ViolationRate)

  if v_total > α then
    if v_total > α_high then
      return Discard, Reason: Critical domain rule violations"
    else
      D_op ← CleansingData(D_op, DR)
    end if
  end if

Step II: Data Quality Assessment

```
for each quality metric DQM_j in DQM do
   score[DQM_j] ← EvaluateQuality(D_op, DQM_j)
   Δ[DQM_j] ← CompareWithReference(D_ref, score[DQM_j])
end for
Δ_total ← Aggregate(Δ)

if Δ_total > β then
   if Δ_total > β_high then
      return Discard, Reason: Severe quality degradation
   else
      D_op ← CleansingData(D_op, DQM)
   end if
end if

Step III: Drift Analysis
   for each feature f of dataset do
      DriftResult.Score[f] ← Distance(D_ref(f) , D_op(f))
      if DriftResult.Score[f] > γ then
         ReportDrift(DriftResult)
         return Discard, Reason: Drift Detected, Go to Model
Retraining
      endif
   end for
   return Accept, Reason: No significant drift, Go to Prediction
```

1) Domain rule validation: This component applies predefined domain-specific rules derived from expert knowledge. The violation rate for each rule DR*i* is computed as:

$$\text{V\_total}() = \frac{number\ of\ records\ violating\ DR_i}{total\ number\ of\ records} \quad (1)$$

The average violation rate V_total is compared against the threshold α. Based on V_total, datasets are classified as accept (< 5% violation), cleansing (5%–15%), or discard (> 15%). Unlike traditional drift detection that analyses whole-dataset distributions, domain rule verification focuses on whether individual key attributes adhere to predefined constraints, enabling early, fine-grained detection of data anomalies.

2) Data quality assessment: A suite of quality metrics (DQM) is computed on D_op. The deviation from D_ref for each metric DQM_j is:

$$\Delta[DQM_j] = |\, DQM_j(D\_op) - DQM_j(D\_ref) \,| \quad (2)$$

where, Δ is the individual quality deterioration per metric.

These deviations are aggregated (via mean or weighted sum) to obtain a cumulative degradation score, Δ_total. Δ_total ← Aggregate(Δ) summarizes the overall quality degradation across all metrics as shown in Eq. (3):

$$\Delta\text{\_total} = \left(\frac{1}{m}\right) \sum \Delta[DQM_j] \quad \text{for } j = 1 \text{ to } m \quad (2)$$

where, *m* is the number of quality metrics.

If Δ_total exceeds a threshold (β), the dataset is refined or discarded. A sudden drop in completeness or spike in redundancy signals anomalies during data acquisition or preprocessing. Datasets classified as Discard at this stage are excluded from training; practitioners must collect or re-generate a compliant dataset before proceeding.

3) Drift analysis: Distributional shifts are monitored using statistical drift detection methods appropriate to each data modality (KS test for tabular data; FID and cosine similarity for images). If the drift score for any feature *f* exceeds threshold γ, the framework triggers model retraining. The per-feature drift score uses the same notation throughout: DriftScore[f] denotes the distributional distance for feature *f* between D_ref and D_op.

C. Model Performance Analysis

Only datasets satisfying all three evaluation criteria proceed to model training or inference. Model performance is continuously monitored; a significant decrease in accuracy or F1-score may indicate a change in the underlying data generation process, inducing model retraining. By conducting data evaluation at the early stages of the ML lifecycle, DcDM reduces the frequency of costly real-time model retraining.

V. Experiments

To validate the effectiveness of the DcDM framework, we conducted experiments using three datasets from different modalities: Electricity Load Diagrams (tabular time-series), CIFAR-10 (image), and VisDA-2017 (image for domain adaptation). For each dataset, multiple operational variants were generated to simulate different types of drift. Each dataset undergoes domain rule evaluation, data quality evaluation, and drift detection sequentially. We then compare model performance before and after refinement, conduct an ablation study to isolate each component's contribution, and compare against established drift detection baselines. All improvements are confirmed to be statistically significant.

A. Experimental Setup

1) Datasets

a) UCI Electricity load diagrams: Time-series dataset capturing electricity consumption by households, used to evaluate time-based drift patterns [29].

b) CIFAR-10: A standard image classification benchmark containing 10 classes of low-resolution (32×32) color images [30].

c) VisDA-2017: A domain adaptation benchmark with synthetic-to-real domain shift in object recognition tasks [31].

2) Experiment Scenario

A systematic experiment scenario was designed as follows:

a) Operational dataset generation: For each dataset, multiple operational variants (e.g., op1, op2) were synthetically generated by applying random transformations to simulate real-world anomalies as shown in Table V. These perturbations were selected to approximate common operational anomalies in each modality while still allowing controlled comparison across datasets.

b) Domain rule evaluation: A customized set of domain-specific rules and threshold values was defined per dataset. Datasets with violation rates below 10% were retained; those between 10%–20% were cleaned; those exceeding 20% were discarded.

TABLE V. SYNTHETIC PERTURBATION AND PARAMETER

Modality	Perturbation	Parameterization (sampled per variant)
Tabular (Electricity)	Missing-value injection	Uniform random masking of 2%–25% of cells, applied independently per column
	Power-surge injection	Additive spikes of 1.5×–3×the column's local rolling mean, inserted at 1%–10% of timestamps
	Usage-pattern shift	Multiplicative rescaling of weekend or off-peak readings by a factor drawn from U(0.5, 2.0)
Image (CIFAR-10 / VisDA-2017)	Brightness distortion	Pixel-wise additive/multiplicative shift, target mean brightness drawn from U(40, 220) on a 0–255 scale
	Gaussian noise injection	σ drawn from U(5, 35) on a 0–255 pixel scale, applied per-channel
	Blurring	Gaussian blur kernel with σ drawn from U(0.5, 3.0)
	Label/channel corruption	Label index reassignment for a sampled fraction (0%–30%) of records; for VisDA-2017, additional RGB channel permutation at a sampled probability

c) Data quality assessment: Quality metrics and thresholds tailored to each dataset type assessed completeness, consistency, and noise levels.

d) Drift analysis: PSI, JS Divergence, KL Divergence, and KS test were applied to tabular datasets; SSIM, PSNR, NIQE, BRISQUE, and FID were used for image datasets.

e) Model performance comparison: Model performance on each refined dataset was compared against its unrefined counterpart.

f) Ablation study: Each pipeline stage (domain rule validation, quality assessment, drift detection) was removed individually to quantify each component's contribution.

g) Baseline comparison: DcDM was compared against ADWIN, DDM, and Evidently AI on the Electricity and CIFAR-10 datasets to contextualize performance gains.

B. Case I: Electricity Load Diagrams

1) Operational dataset generation: Ten operational datasets (op1_elec to op10_elec) were generated from the electricity dataset, simulating anomalies including missing values, power surges, and unusual usage patterns.

2) Domain rule evaluation: Domain rules and thresholds for the Electricity dataset are defined in Table VI.

TABLE VI. DOMAIN RULES AND THRESHOLDS (ELECTRICITY)

NO	Rule	Threshold	Description
DR1	Power ≥ 0	≥ 5%	Power values must be non-negative
DR2	0 ≤ Valid Power ≤ 5000kW	≥ 10%	Valid power values within the operating range
DR3	Weekend power < weekday avg.	< 500	Weekend power usage must be lower than the weekday average
DR4	Hourly change rate	≥ 15%	Hourly change rate must be within ±20%

The domain rule evaluation results are presented in Table VII. Datasets receiving a 'Cleansing' judgment had violating records removed before proceeding to quality assessment.

TABLE VII. DOMAIN RULE EVALUATION (ELECTRICITY)

Dataset	DR Violations	Judgment	New Dataset
op1_elec	DR1: 2%, DR4: 18%	Cleansing	op1-1_elec
op2_elec	DR2: 12%	**Discard**	-
op3_elec	DR3 Violation	Cleansing	op3-1_elec
op4_elec	DR4: 25%	Cleansing	op4-1_elec
op5_elec	DR2: 5%, DR4: 10%	Keep	
op6_elec	DR1: 7%	**Discard**	-
op7_elec	DR4: 16%	Cleansing	op7-1_elec
op8_elec	DR3 Violation, DR4: 22%	Cleansing	op8-1_elec
op9_elec	DR2: 18%	**Discard**	-
op10_elec	DR4: 14%	Keep	

3) Data quality assessment: Quality thresholds for the Electricity dataset: Completeness ≥ 95%, Consistency ≥ 98% (no duplicate timestamps), Timeliness ≥ 97% (records within expected intervals). Results are presented in Table VIII.

TABLE VIII. DATA QUALITY ASSESSMENT (ELECTRICITY)

Dataset	Comp	Cons	Time	Judgment	New Dataset
op1-1_elec	94%	99%	98%	Cleansing	op1-2_elec
op3-1_elec	96%	98%	99%	Keep	
op4-1_elec	98%	97%	97%	Keep	
op5_elec	99%	98%	96%	Keep	op5_elec
op7-1_elec	95%	96%	94%	Cleansing	op7-2_elec
op8-1_elec	93%	94%	97%	Cleansing	op8-2_elec
op10_elec	99%	99%	98%	Keep	

4) Drift analysis: The KS test was used to detect drift between refined operational datasets and the original data. As shown in Fig. 3, a p-value below 0.05 indicates drift. Most unrefined datasets (op1_elec, op3_elec, op4_elec, op7_elec, op8_elec) showed significant drift, while their refined counterparts showed substantially lower drift probability. The op5_elec dataset exhibited borderline drift (p-value = 0.039) despite passing domain rule and quality checks, triggering model retraining.

5) Model performance comparison: Using a Random Forest classifier, average model accuracy improved from 0.821 to 0.869, and F1-score increased from 0.824 to 0.872, representing relative improvements of 5.85% and 5.83%, respectively. Fig. 4 shows the model performance on the Electricity operational datasets.

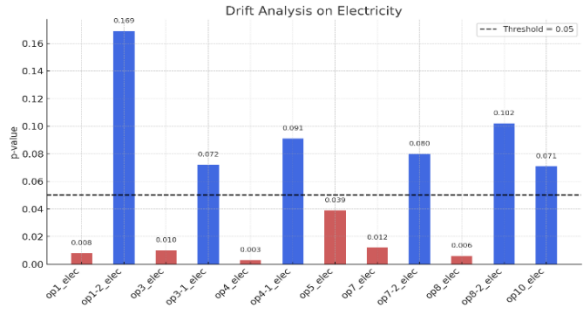

Fig. 3. Drift analysis (electricity).

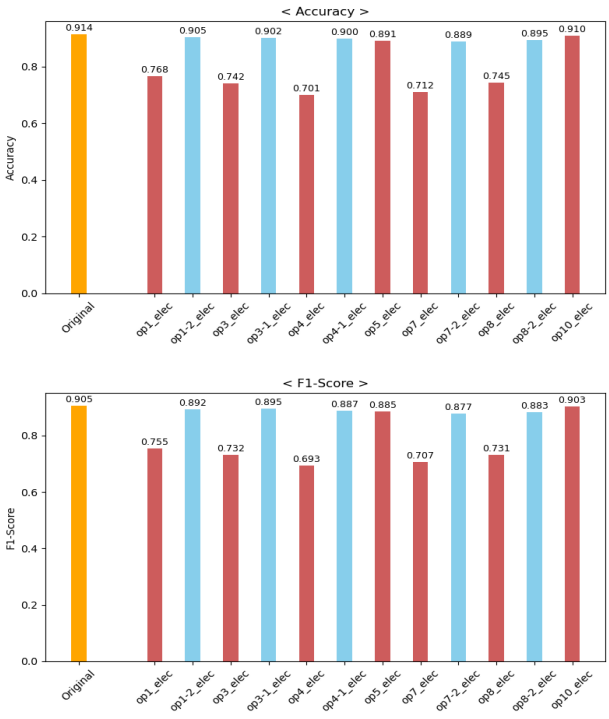

Fig. 4. Model performance (electricity).

C. Case II: CIFAR-10

1) Operational datasets generation: Ten CIFAR-10 operational datasets (op1_CIFAR-10 to op10_CIFAR-10) were generated to simulate brightness distortion, noise injection, blurring, and label mismatches.

2) Domain rule evaluation: The domain rules of the CIFAR-10 dataset are defined in Table IX, based on human visual quality perception and ML sensitivity thresholds.

TABLE IX. DOMAIN RULES AND THRESHOLDS (CIFAR-10)

NO	Rule	Threshold	Description
DR1	Brightness	80 < brightness < 180	Image brightness level within the perceptible range
FDR2	Blur	Sharpness score < 100	Image sharpness sufficient for recognition
DR3	Noise	Noise level < 30	Maximum acceptable noise level
DR4	Label consistency	No corrupted label values	All labels must be valid class indices (0–9)

TABLE X. DOMAIN RULE EVALUATION (CIFAR-10)

Dataset	DR Violations	Judgment	New Dataset
op1_ CIFAR-10	DR1(Brightness): 18%	Cleansing	op1-1
op2_ CIFAR-10	DR3(Noise): 25%	**Discard**	-
op3_ CIFAR-10	DR2(Blur): 12%	Cleansing	op3-1
op4_ CIFAR-10	DR1: 5%, DR2: 17%	Cleansing	op4-1
op5_ CIFAR-10	DR4(Label mismatch): 30%	**Discard**	-
op6_ CIFAR-10	DR1: 2%, DR3: 10%	Keep	
op7_ CIFAR-10	DR2: 9%	Keep	
op8_ CIFAR-10	DR3: 4%, DR4: 21%	Cleansing	op8-1
op9_CIFAR-10	DR1: 0%, DR2: 0%	Keep	
op10_CIFAR-10	DR1: 6%, DR2: 6%, DR3: 6%	Cleansing	op10-1

3) Data quality assessment: For CIFAR-10, perceptual quality metrics appropriate to 32×32 low-resolution images are applied (Table XI). PSNR, SSIM, NIQE, and BRISQUE thresholds are calibrated to the native resolution of CIFAR-10 images, ensuring that the quality criteria reflect realistic expectations for the target modality rather than high-resolution benchmarks (Table XII).

TABLE XI. QUALITY-SPECIFIC DATA REQUIREMENT (CIFAR-10)

NO	Quality Metric	Threshold	Description
DQ1	PSNR	≥ 28 dB	Peak Signal-to-Noise Ratio (applicable to 32×32 low-res images with adjusted baseline)
DQ2	SSIM	≥ 0.9	Structural Similarity Index
DQ3	NIQE	≤ 5.0	Naturalness Image Quality Evaluator
DQ4	BRISQE	≤ 40	Blind/Referenceless Image Spatial Quality Evaluator

TABLE XII. DATA QUALITY ASSESSMENT (CIFAR-10)

Dataset	DQ1	DQ2	DQ3	DQ4	Judgment	New Dataset
op1-1	26.5	0.91	5.5	45	Cleansing	op1-2
op3-1	29.1	0.93	4.9	39	Keep	-
op4-1	27.0	0.9	5.3	43	Cleansing	op4-2
op6	30.2	0.95	3.5	30	Keep	-
op7	31.5	0.96	3.2	28	Keep	-
op8-1	24.7	0.88	6.2	52	Cleansing	op8-2
op9	32.1	0.97	2.8	25	Keep	-
op10-1	25.9	0.89	5.9	47	Cleansing	op10-2

4) Drift analysis: FID (Fréchet Inception Distance) was applied to datasets passing the quality assessment step (Fig. 5). A threshold of FID > 40 denotes significant drift. Refined datasets (e.g., op1-2_CIFAR-10) exhibited substantially reduced FID compared to their unrefined counterparts (e.g., op1_CIFAR-10).

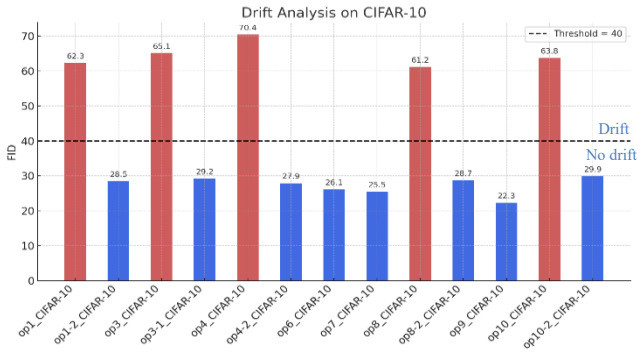

Fig. 5. Drift analysis (CIFAR-10).

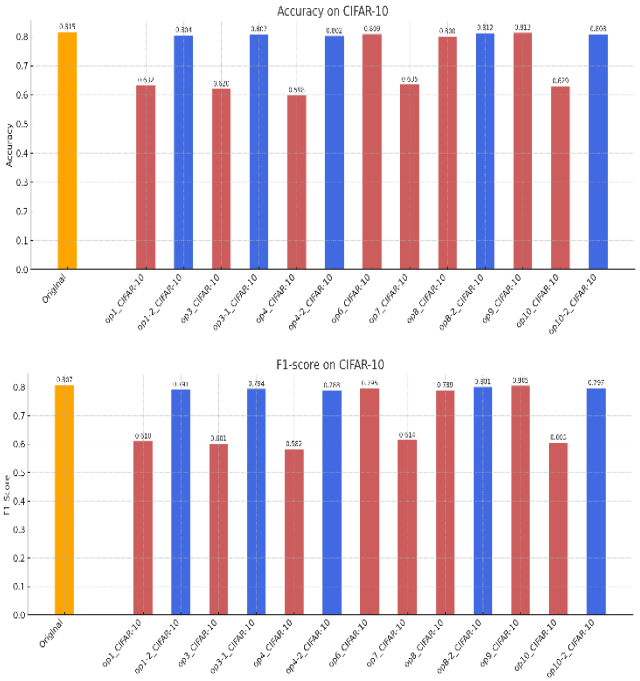

Fig. 6. Model performance (CIFAR-10).

5) Model performance comparison: Fig. 6 shows the model performance on the CIFAR-10 operational datasets. Average accuracy improved from 0.723 to 0.770, and F1-score improved from 0.726 to 0.774, representing improvements of 6.50% and 6.61%, respectively.

D. Case III: VisDA-2017

1) Operational dataset generation: Ten synthetic variants of the VisDA-2017 dataset (op1_Vis to op10_Vis) were generated to reflect image resolution shifts, channel misalignment, brightness distortion, and label noise.

2) Domain rule evaluation: Domain rules: DR1 (Image resolution), DR2 (RGB channel alignment), DR3 (Semantic label validity), DR4 (Brightness). Evaluation results are presented in Table XIII.

3) Data quality assessment: The same image quality metrics as CIFAR-10 (Table X) were applied, adjusted to VisDA-2017's higher-resolution inputs. Results are presented in Table XIV.

TABLE XIII. DOMAIN RULES EVALUATION (VISDA-2017)

Dataset	DR Violations	Judgment	New Dataset
op1_Vis	DR1(Resolution): 16%, DR2(Channels): 0%	Cleansing	op1-1_Vis
op2_Vis	DR3(Label mismatch): 28%	**Discard**	-
op3_Vis	DR4(Brightness): 11%	Cleansing	op3-1_Vis
op4_Vis	DR1: 5%, DR4: 21%	Cleansing	op4-1_Vis
op5_Vis	DR2: 12%	**Discard**	-
op6_Vis	DR1: 0%, DR4: 0%	Keep	op6_Vis
op7_Vis	DR3: 18%	Cleansing	op7-1_Vis
op8_Vis	DR4: 26%	Cleansing	op8-1_Vis
op9_Vis	DR2: 5%	Keep	op9_Vis
op10_Vis	DR1: 3%, DR2: 6%, DR3: 4%	Cleansing	op10-1_Vis

TABLE XIV. DATA QUALITY ASSESSMENT (VISDA-2017)

Dataset	DQ1	DQ2	DQ3	DQ4	Judgment	New Dataset
op1-1_Vis	27.4	0.89	5.2	44	Cleansing	op1-2_Vis
op3-1_Vis	25.1	0.87	5.9	49	Cleansing	op3-2_Vis
op4-1_Vis	26.5	0.88	5.4	47	Cleansing	op4-2_Vis
op6_Vis	29.8	0.92	3.7	31	Keep	
op7-1_Vis	26.2	0.89	5.6	46	Cleansing	op7-2_Vis
op8-1_Vis	24.0	0.85	6.4	53	Cleansing	op8-2_Vis
op9_Vis	30.1	0.93	3.2	28	Keep	
op10-1_Vis	25.8	0.86	5.7	50	Cleansing	op10-2_Vis

4) Drift analysis: FID was employed to detect distributional drift, with a threshold of FID > 40 denoting significant drift. Refined datasets demonstrated consistently lower FID scores compared to their unrefined counterparts (Fig. 7).

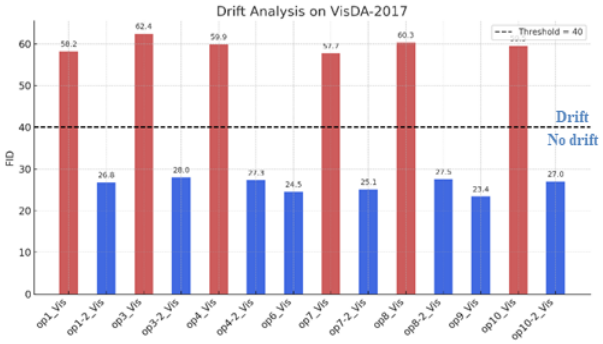

Fig. 7. Drift analysis (VisDA-2017).

5) Model performance comparison: By applying domain rule validation and quality assessment, accuracy improved from 0.568 to 0.639 and F1-score from 0.574 to 0.642, corresponding to improvements of 12.5% and 11.85%, respectively—the largest gains across all three datasets, reflecting the challenging synthetic-to-real distribution gap that DcDM's semantic validation effectively addresses (Fig. 8).

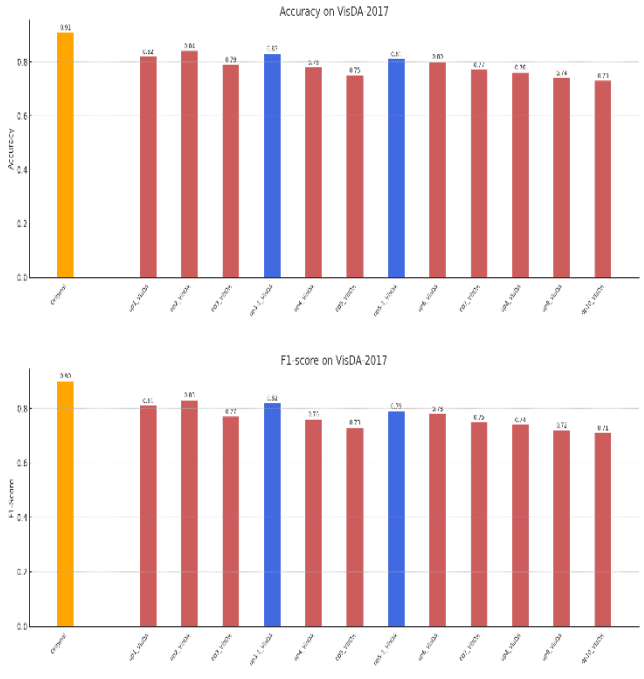

Fig. 8. Model performance (VisDA-2017).

E. Result Analysis

1) Statistical significance analysis: Paired t-tests were applied to the accuracy and F1-scores between refined and unrefined datasets. Across all three datasets, improvements were statistically significant at the 1% level ($p < 0.01$). Table XV summarizes the statistical significance of each dataset.

TABLE XV. STATISTICAL SIGNIFICANCE OF PERFORMANCE IMPROVEMENTS

Dataset	Metric	Baseline (Mean ± CI)	DcDM Refined (Mean ± CI)	p-value
Electricity	Accuracy	0.821 ± 0.012	0.869 ± 0.010	< 0.01
	F1-score	0.824 ± 0.013	0.872 ± 0.009	< 0.01
CIFAR-10	Accuracy	0.723 ± 0.015	0.770 ± 0.014	< 0.01
	F1-score	0.726 ± 0.016	0.774 ± 0.013	< 0.01
VisDA-20	Accuracy	0.568 ± 0.020	0.639 ± 0.018	< 0.01
	F1-score	0.574 ± 0.019	0.642 ± 0.017	< 0.01

Effect sizes (Cohen's $d > 0.6$) confirm that DcDM consistently yields moderate-to-strong gains over baseline methods, reinforcing the reliability of the reported results.

2) Ablation study: To isolate the contribution of each DcDM component, we conducted a controlled ablation study on the Electricity and CIFAR-10 datasets. We evaluated five configurations: no refinement (baseline), domain rule validation only, quality assessment only, domain rule + quality assessment (without drift detection), and the full DcDM pipeline. Results are presented in Table XVI.

The ablation results confirm that each component contributes incrementally to overall performance. Domain rule validation alone yields a moderate improvement of approximately 2.2–2.5% in accuracy by removing semantically invalid samples. Quality assessment alone provides a slightly larger gain (approximately 3.0–3.4%) by filtering low-quality data. Combining both without drift detection achieves near-full performance recovery, with an improvement of approximately 3.7–4.0%. The full DcDM pipeline, which additionally incorporates drift-triggered retraining, achieves the largest gains, confirming that all three components are necessary for optimal performance.

TABLE XVI. ABLATION STUDY: CONTRIBUTION OF EACH PIPELINE COMPONENT

Configuration	Electricity		CIFAR-10		VisDA	
	Acc.	F1	Acc.	F1	Acc.	F1
No refinement (Baseline)	0.821	0.824	0.723	0.726	0.568	0.574
Domain Rule Only	0.843	0.846	0.748	0.751	0.592	0.595
Quality Assessment Only	0.851	0.854	0.757	0.760	0.608	0.611
Domain Rule + Quality (DcDM without drift detection)	0.858	0.861	0.763	0.767	0.624	0.627
Full DcDM (All three steps)	0.869	0.872	0.770	0.774	0.639	0.642

3) Comparison with existing drift detection baselines: To contextualize DcDM's performance gains, we compared it against three established drift detection approaches: ADWIN [19], DDM [16], and Evidently AI [27], applied to the Electricity and CIFAR-10 datasets. All baselines operate reactively (detecting drift post-deployment) and do not incorporate semantic rule validation or data quality gating prior to training. Results are presented in Table XVII.

TABLE XVII. COMPARISON WITH REACTIVE DRIFT DETECTORS

Method \ Dataset	Electricity		CIFAR-10	
	Acc.	F1	Acc.	F1
No refinement	0.821	0.824	0.724	0.726
ADWIN	0.834	0.836	0.737	0.739
DDM	0.839	0.841	0.741	0.743
Evidently AI	0.845	0.847	0.748	0.751
DcDM (Ours)	0.869	0.872	0.770	0.774

The comparison should not be interpreted as a direct algorithmic competition because ADWIN, DDM, and Evidently AI were originally designed as reactive post-deployment monitoring solutions. The purpose of this comparison is to illustrate the practical difference between proactive data-centric prevention and reactive drift detection paradigms. ADWIN and DDM, which focus on detecting concept drift in streaming data without upstream data quality intervention, achieve modest improvements over unrefined data (approximately 1.3–1.8% accuracy gain). Evidently, AI, which monitors distributions at the feature level, provides slightly larger gains (approximately 2.4–2.7%), but still falls short of DcDM's 5.8–6.5% improvements. These results confirm that DcDM's proactive, pre-training integration of semantic validation and quality

assessment provides complementary and superior benefits over reactive monitoring approaches.

4) Sensitivity analysis on thresholds: The sensitivity analysis on thresholds (α for domain rules, β for quality metrics, γ for drift detection) is presented in Table XVIII. Performance remained stable with accuracy and F1-score fluctuating within ±2% under ±10% threshold variation, demonstrating robustness to threshold misconfiguration. Under extreme threshold variations of ±30%, performance degradation remained bounded below 6%.

TABLE XVIII. SENSITIVITY ANALYSIS ON THRESHOLDS

Threshold Setting	Accuracy (Mean)	F1-score (Mean)	Observation
α=5%, β=5%, γ=0.05	0.861	0.865	Conservative: fewer data passes; safe but slightly lower coverage
α=10%, β=10%, γ=0.05	0.869	0.872	Optimal balance: best accuracy/F1 combination across all datasets
α=20%, β=20%, γ=0.1	0.854	0.858	Slight degradation: more borderline data allowed through
α=30%, β=30%, γ=0.1	0.835	0.840	Clear degradation: lenient thresholds reduce filtering effectiveness

5) Overhead and computing cost: All experiments were conducted on a local workstation equipped with an Intel Core i7 processor, 32 GB of RAM, and an NVIDIA RTX 3080 GPU. Average processing time per step is summarized in Table XIX.

TABLE XIX. OVERHEAD AND COMPUTING COST

Dataset	Domain Rule Validation	Quality Assessment	Drift Detection
Electricity (100K records)	2.1 sec	3.4 sec	1.9 sec
CIFAR-10 (50K images)	3.2 sec	4.8 sec	2.7 sec
VisDA-2017 (50K images)	3.5 sec	5.0 sec	3.0 sec

Batch-level processing time for the Electricity dataset: domain rule validation $\approx$ 12 ms per 1,000 records, quality assessment ≈ 18 ms, drift detection (KS test) ≈ 9 ms. For image datasets, FID-based drift detection averaged 0.04 sec per batch of 500 images. These results indicate minimal computational overhead suitable for batch processing. For large-scale streaming environments, a two-level validation strategy is envisioned: 1) lightweight domain rules and quality checks in real time, and 2) comprehensive drift analysis asynchronously in batch mode.

We also measured batch-level processing time to evaluate the practicality of the framework. For the Electricity dataset, domain rule validation required approximately 12 ms per batch of 1,000 records, quality assessment required 18 ms, and drift detection (KS test) required 9 ms. Similar magnitudes were observed for image datasets, where FID-based drift detection averaged 0.04 sec per batch of 500 images, which corresponds to the reported totals for 50K images. These results indicate that the framework introduces minimal computational overhead and is therefore feasible for batch processing. For large-scale streaming data, we envision a two-level validation strategy: 1) lightweight domain rules and quality checks applied in real time at the high-velocity streaming environments, and 2) comprehensive validation performed asynchronously in batch mode.

VI. DISCUSSION

The DcDM framework demonstrates notable improvements in robustness and reliability by proactively addressing data drift. The main contributions include the integration of domain validation, statistical drift analysis, and automation in machine learning pipelines. Nevertheless, this study has several limitations.

1) Use of synthetic drift generation: All operational datasets were generated via synthetic perturbations. While this enables controlled experimentation, synthetic drift may not fully capture the complexity of real-world drift phenomena, such as non-linear, gradual, or concept-intertwined drift in industrial sensor networks or healthcare data streams. Future validation on real deployment logs or domain-specific streaming datasets (e.g., medical imaging or financial transactions) is necessary to confirm robustness under realistic drift conditions.

2) Reliance on domain experts for rule definition: The current framework relies on domain experts to define semantic rules, which can be costly and domain-specific. Automated rule induction through data profiling, pattern mining, or meta-learning could complement expert-driven rules. For example, profiling tools can automatically infer valid ranges, correlations, and label consistency from historical data, reducing human dependency in rapidly evolving domains.

3) Fixed thresholds: Although the sensitivity analysis confirmed robustness under moderate variation, fixed thresholds may not adapt optimally across all deployment scenarios. A promising direction is feedback-driven or reinforcement learning-based threshold adaptation, allowing DcDM to dynamically balance false alarms and missed drift detections as data distributions evolve.

4) No validation against real deployment environment: Although real-world deployment datasets with verified drift annotations were not available for this study, we attempted to approximate realistic operational conditions by applying perturbations derived from commonly reported degradation patterns, including sensor noise, missing values, illumination changes, image blur, and label inconsistencies. These perturbations represent typical causes of data-quality degradation reported in industrial monitoring, computer vision, and data-streaming applications. Future work will evaluate DcDM using naturally evolving datasets collected from production environments to further validate its practical effectiveness.

VII. CONCLUSION AND FUTURE WORK

This study proposed a proactive, data-centric methodology, DcDM (Data-centric Drift Management), for managing data drift across the ML lifecycle. Unlike traditional model-centric approaches that respond to performance degradation, DcDM

emphasizes early intervention by evaluating datasets before model training. The proposed framework systematically integrates three components: 1) domain rule validation to assess semantic correctness, 2) quality assessment based on goal-driven metrics, and 3) statistical drift detection to quantify distributional shifts.

By applying this methodology across diverse datasets—Electricity Load Diagrams, CIFAR-10, and VisDA-2017—we demonstrated performance improvements of 6%–12.5% in both accuracy and F1-score. An ablation study confirmed that each pipeline component contributes incrementally to overall gains, and a baseline comparison against ADWIN, DDM, and Evidently AI confirmed that DcDM's proactive, pre-training integration provides superior benefits over reactive monitoring approaches. Statistical significance tests ($p < 0.01$, Cohen's $d > 0.6$) and sensitivity analyses confirm the robustness of these results.

Future work will focus on: 1) validating DcDM on real-world deployment logs and natural drift scenarios; 2) automating domain rule induction via data profiling and meta-learning to reduce expert dependency; 3) developing feedback-driven, adaptive threshold mechanisms; 4) extending DcDM to explicitly handle concept drift; and 5) integrating DcDM into production-grade MLOps pipelines with privacy-preserving rule management.

ACKNOWLEDGMENT

This work was supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP)-Innovative Human Resource Development for Local Intellectualization program grant funded by the Korea government(MSIT)(IITP-2026-RS-2022-00156334).

REFERENCES

[1] C. Zhang, Y. Bengio, and M. Hardt, “Understanding deep learning requires rethinking generalization,” arXiv preprint arXiv.1611.03530, https://doi.org/10.48550/arXiv.1611.03530.

[2] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, “A survey on concept drift adaptation,” ACM Comput. Surv., vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014, doi: 10.1145/252381.

[3] A. Tsymbal, “The problem of concept drift: Definitions and related work,” Computer Science Department, Trinity College Dublin, Tech. Rep. TCD-CS-2004-15, 2004.

[4] R. Baier, F. Heine, and F. Puppe, “Handling concept drift in process mining,” Expert Systems with Applications, vol. 128, pp. 60–77, 2019.

[5] S. Wang, M. Minku, and X. Yao, “A review of online class imbalance learning: Trends and challenges,” Progress in Artificial Intelligence, vol. 5, no. 3, pp. 123–135, 2016.

[6] A. Webb, D. Page, and T. A. Miller, “A real-time approach to concept drift detection and model refresh in streaming data,” in Proc. ACM Int. Conf. Information and Knowledge Management (CIKM), 2020.

[7] A. Breck, S. Cai, E. Nielsen, J. Salib, and D. Sculley, “The ML test score: A rubric for ML production readiness and technical debt reduction,” in Proc. SysML Conf., 2019.

[8] A. Bontempelli et al., “Data-centric AI: A comparative review,” Data-Centric AI Workshop at NeurIPS, 2021.

[9] A. Zoller, and M. Huber, “Benchmarking the robustness of machine learning models to data quality issues,” arXiv preprint, arXiv:2202.04643, 2022.

[10] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, “Learning under concept drift: A review,” IEEE Trans. Knowl. Data Eng., vol. 31, no. 12, pp. 2346–2363, Dec. 2019.

[11] Moez Ali, “Understanding data drift and model drift: Drift detection in Python,” DataCamp, 2023. [Online]. Available: https://www.datacamp.com/tutorial/understanding-data-drift-model-drift. [Accessed: Jun. 2025].

[12] B. Yurdakul and J. Naranjo, "Statistical properties of populations stability index", Journal of Risk Model Validation, vol. 14, no. 4, pp.89-100, 2020.

[13] S. Kullback, and R. A. Leibler, “On information and sufficiency”, The Annals of Mathematical Statistics, vol. 22, no. 1, pp.79-86, 1951.

[14] F. Nielsen, “On a generalization of the Jensen–Shannon divergence and the Jensen–Shannon centroid”, Entropy, vol. 22, no. 2, Art. No. 221, 2020.

[15] S. Ashok, S. Ezhumalai, T. Patwa, "Remediating data drift and re-establishing ML models", Procedia Computer Science, vol. 218, pp.799-809, 2023.

[16] I. Žliobaitė, “Learning under concept drift: Early drift detection method,” *in* Lect. Notes Comput. Sci. (LNAI 3171), pp. 286–295, 2004, https://doi.org/10.1007/978-3-540-28645-5_29.

[17] Manuel Baena-García, José Campo-Ávila, Raúl Fidalgo-Merino, Albert Bifet, Ricard Gavald, and Rafael Morales-Bueno. “Early drift detection method,” in Proceedings of the International Workshop on Knowledge Discovery from Data Streams, pp.77–86. 2006.

[18] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, “Exponentially weighted moving average charts for detecting concept drift,” Pattern Recognit. Lett., vol. 33, no. 2, pp. 191–198, Dec. 2012, https://doi.org/10.48550/arXiv.1212.6018

[19] A. Bifet and R. Gavaldà, “Learning from time-changing data with adaptive windowing,” in Proc. SIAM Int. Conf. Data Mining, 2007, pp. 443–448, https://doi.org/10.1137/1.9781611972771.42

[20] I. Frías-Blanco, J. d. Campo-Ávila, G. Ramos-Jiménez, R. Morales-Bueno, A. Ortiz-Díaz, and Y. Caballero-Mota, “Online and non-parametric drift detection methods based on Hoeffding’s bounds,” IEEE Trans. Knowl. Data Eng., vol. 27, no. 3, pp. 810–823, Mar. 2015, https://doi.org/10.1109/TKDE.2014.2345382

[21] A. Gretton, et al., “A kernel two-sample test”. Journal of Machine Learning Research, vol. 13, pp.723-773, 2012.

[22] Min Jin Chong and David Forsyth, “Effectively unbiased FID and inception score and where to find them”, in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit(CVPR) 2020, https://doi.org/10.48550/arXiv.1911.07023

[23] B. Celik and J. Vanschoren, “Adaptation strategies for automated machine learning on evolving data”, IEEE Transactions on Pattern Analysis and Machine Intelligence“, vol.43, no.9, pp. 3067-3078, 2021, https://doi.org/10.1109/TPAMI.2021.3062900

[24] E. Kalaitzidis et al., “AML4S: An AutoML pipeline for data data streams”, MDPI Machine learning & Knowledge extraction, 7(3), pp1-33, July. 2025, https://doi.org/10.3390/make7030087

[25] M. Schaller et al., “AutoML for multi-class anomaly compensation of sensor drift”, arXiv preprint arXiv:2502.19180, 2025, https://doi.org/10.48550/arXiv.2502.19180

[26] A. Ng, "A chat with Andrew on MLOps: From model-centric to data-centric AI", DeepLearningAI, 2021, [Online]. Available: https://duly.kr/4bfuMF5

[27] E. Breck et al., "Evidently AI: Open-source ML monitoring and testing," GitHub repository, 2022. [Online]. Available: https://github.com/evidentlyai/evidently

[28] J. Van Looveren et al., "Alibi Detect: Algorithms for outlier, adversarial and drift detection," J. Mach. Learn. Res., 2022.

[29] UCI Electricity Load Diagrams Dataset. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams. [Accessed: Jun. 2025].

[30] CIFAR-10 Dataset. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: Jun. 2025]

[31] VisDA-2017 Dataset. [Online]. Available: https://github.com/VisionLearningGroup/visda-2017 [Accessed: Jun. 2025]