The Journal of Systems and Software 217 (2024) 112184

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

In Practice

Adaptive data quality scoring operations framework using drift-aware mechanism for industrial applications ☆

Firas Bayram [a], Bestoun S. Ahmed [a,b,*], Erik Hallin [c]

[a] Department of Mathematics and Computer Science, Karlstad, 651 88, Sweden
[b] Department of Computer Science, Faculty of Electrical Engineering, Czech Technical University, Prague, Czech Republic
[c] Uddeholms AB, Hagfors, 683 33, Värmlands län, Sweden

ARTICLE INFO

Keywords:
Industrial application
Adaptive data quality
Data quality scoring
Data validation
Data-driven AI
Drift detection

ABSTRACT

Within data-driven artificial intelligence (AI) systems for industrial applications, ensuring the reliability of the incoming data streams is an integral part of trustworthy decision-making. An approach to assess data validity is data quality scoring, which assigns a score to each data point or stream based on various quality dimensions. However, certain dimensions exhibit dynamic qualities, which require adaptation on the basis of the system's current conditions. Existing methods often overlook this aspect, making them inefficient in dynamic production environments. In this paper, we introduce the Adaptive Data Quality Scoring Operations Framework, a novel framework developed to address the challenges posed by dynamic quality dimensions in industrial data streams. The framework introduces an innovative approach by integrating a dynamic change detector mechanism that actively monitors and adapts to changes in data quality, ensuring the relevance of quality scores. We evaluate the proposed framework performance in a real-world industrial use case. The experimental results reveal high predictive performance and efficient processing time, highlighting its effectiveness in practical quality-driven AI applications.

1. Introduction

Industries and businesses are actively accumulating data in unprecedented volumes, marking a definitive shift towards a data-centric paradigms (Reis and Saraiva, 2021). Within this landscape, data has transcended its conventional role to become the cornerstone of success for artificial intelligence (AI) software solutions. Data holds an intrinsic value due to its inseparable link to the life cycle of machine learning (ML), which constitutes the primary type of AI. Therefore, assessing the quality of the collected data becomes an essential and imperative aspect in building robust and reliable ML solutions (Hazen et al., 2014). Remarkably, a prevalent concern in the industry is the disproportionate allocation of efforts in ML projects in research. According to a recent study conducted by MIT scientists (Stonebraker and Rezig, 2019), research institutions often allocate 90% of their efforts to improve ML algorithms and only 10% to data preparation and validation. The authors suggested that these numbers should be reversed for better overall outcomes. In light of this, the significance of data quality (DQ) assurance extends beyond mere procedural correctness. It plays a pivotal role in influencing the cost-effectiveness and operational efficiency of industrial processes and operations.

There are two main approaches to data quality assessment: quantitative assessments that involve scales and metrics to quantify aspects of data quality, and qualitative assessments that focus on inherent characteristics and subjective evaluations (Zaveri et al., 2016). Quantitative assessments provide a measurement, offering more detailed information about the objective estimates of data quality. This approach involves assigning specific numerical values to various aspects, called a quality score or index (Chen et al., 2014), which enables a quantifiable understanding of data quality. In contrast, qualitative assessments contribute to a more holistic perspective by exploring the intrinsic qualities and subjective aspects of the data, referred to as data profiling (Liu and Özsu, 2009). Through the use of scales and metrics, our study focuses on quantitative data scoring, as it provides valuable insights for industries.

Data quality scoring is a methodological approach that involves evaluating and assigning scores to data records based on predefined criteria across several data quality dimensions (Batini et al., 2009). Each data quality dimension captures a unique aspect, collectively contributing to a thorough assessment of the overall data quality. The scores indicate the acceptance level of the addressed data quality

☆ Editor: Daniel Mendez.

* Corresponding author at: Department of Mathematics and Computer Science, Karlstad, 651 88, Sweden.
E-mail addresses: firas.bayram@kau.se (F. Bayram), bestoun@kau.se (B.S. Ahmed), erik.hallin@uddeholm.com (E. Hallin).

https://doi.org/10.1016/j.jss.2024.112184
Received 29 February 2024; Received in revised form 9 August 2024; Accepted 13 August 2024
Available online 14 August 2024

dimensions, distinguishing between high- and poor-quality data. Furthermore, data quality scores are interpreted as the degree to which data quality conforms to specified aspects of data quality (Taleb et al., 2021). In the context of large-scale systems, the importance of these data quality scores extends beyond the evaluation of data quality and can be exploited to enhance the overall performance of the ML system (Chen et al., 2021).

From a data-centric AI perspective, assessing data quality is crucial; nevertheless, it adds computational costs to systems, especially in real-time production environments. The practicality of these assessments in operational contexts poses a notable challenge due to inherent complexities and resource-intensive nature in industrial use cases. Therefore, streamlining these processes becomes a vital requirement to ensure their effectiveness and success in practical applications (Budach et al., 2022). Two primary approaches exist for data quality scoring: a traditional standard approach and an automated approach utilizing an ML-based system (Widad et al., 2023). The standard approach involves checking a predefined set of data quality dimensions and assigning scores to each dimension. In contrast, ML-based methods employ an ML regressor to predict the score of the processed data window instead of the manual scoring procedure.

In the domain of industrial processes where real-time data analysis is crucial, ML-based methods present benefits in terms of efficiency and speed (Wu et al., 2019). Using ML algorithms can significantly reduce the time and resources required for data quality assessment, especially in large-scale environments (Bayram et al., 2023a). The choice of an ML-based approach over traditional methods is driven by the need for a more scalable, efficient, and adaptive solution. ML-based methods can process large volumes of data quickly in real-time, providing a more timely evaluation of data quality. The predictive capabilities of ML models enable faster analysis and decision-making, enhancing the overall effectiveness of data quality evaluation processes. This capability is particularly beneficial in industries such as manufacturing, where early identification and resolution of data quality issues can optimize production processes and minimize downtime.

ML-based approaches can be categorized into adaptive and non-adaptive methods (Clerc et al., 2016). Regarding data quality assessment, adaptivity is not only about ML model adjustments. Rather, it refers to the ability to respond effectively to dynamic changes in data quality dimensions. These adaptive methods continuously monitor and update their quality assessment criteria based on incoming data, allowing them to respond effectively to evolving conditions. In contrast, non-adaptive methods rely on static quality assessment criteria throughout the analysis, potentially overlooking changes in data quality over time.

Existing data quality scoring frameworks often neglect the crucial aspect of adaptivity, presenting a major obstacle to effective management of dynamic data quality. Specifically, non-adaptive ML frameworks have two inherent limitations that need to be addressed. Firstly, determining the optimal size of the optimal time checkpoint to retrain the ML model in production can be challenging and may lead to inefficiencies. Secondly, certain data quality dimensions exhibit a dynamic nature, reflecting the fluctuating conditions of industrial systems. For example, highly relevant data in one phase may be regarded as of a lower quality under different circumstances. The limitations of non-adaptive frameworks highlight the importance of retraining the ML model to capture these dynamic changes. ML models are trained on historical data to learn the characteristics of high- and low-quality data according to current conditions. When data characteristics change, quality scores also change, and an adaptation signal should be triggered to adapt the ML model to the evolving data quality characteristics, ensuring constant learning for more accurate assessments.

This paper introduces an innovative approach to tackle the dynamic challenges of data quality assessment by proposing a novel framework that integrates adaptivity into ML-based data quality scoring methodologies. Our main contribution lies in addressing the evolving nature of data quality and incorporating drift detection mechanisms to enhance data quality scoring accuracy. The proposed framework dynamically adjusts the retraining process based on evolving data patterns, enabling a more precise and adaptable data quality assessment over time. In addition, we introduce adaptive mechanisms to facilitate dynamic responses and re-calibration of data quality scores according to the prevailing system conditions. This approach effectively meets the demands of large-scale industrial processes where data quality requirements evolve continuously.

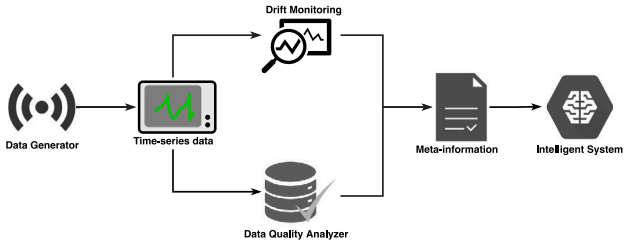


Fig. 1. Quality-driven data workflow for informed decision-making in industrial applications.

The remainder of the paper is structured as follows. Section 2 outlines the conceptual background necessary to understand our proposed framework. In Section 3, we provide an overview of the existing literature related to data quality assessments. Following this, Section 4 introduces the framework with its development and deployment phases. Section 5 presents implementation details along with an analysis of its predictive performance and processing time efficiency. Finally, in Section 6, we summarize the key contributions of the paper and discuss potential opportunities for future research.

2. Conceptual background

Continuous monitoring and assessment of the incoming data streams are crucial to ensure that dynamic changes are captured and that the prevalent condition (concept) remains consistent. In data-centric applications, it is essential to verify that data quality remains high. Data quality is important in ensuring the data remains useful, as low-quality data can lead to physical failures or inaccuracies in sensor readings (Teh et al., 2020). This process is illustrated in Fig. 1, where time-series data collected from the data source undergoes both drift detection and data quality assessment. Consequently, the AI system benefits from valuable meta-information obtained from these stages. Continuous drift monitoring ensures that the incoming data remain contextually similar to the training data (Polyzotis et al., 2018), allowing intelligent systems to make informed decisions based on high-quality information (Zha et al., 2023).

Within the context of data-driven applications, these changes can have implications for the performance of both data quality and application-level ML systems. They signify the need for adaptation to maintain system reliability. In practice, application-level ML systems utilize drift detection to sustain performance in dynamic environments, while data-quality scoring ML systems use it to ensure the quality and reliability of the generated data. Despite their different objectives, both application-level and data quality scoring ML systems rely on the same core principle: monitoring changes in data characteristics. This common foundation allows for the successful application of identical drift detection mechanisms across both model types. This paper focuses primarily on adapting data quality scoring ML systems within the holistic AI application. This section provides essential background knowledge on these core concepts, laying the foundation for a comprehensive understanding of the fundamental elements within the scope of this research.

2.1. Drift monitoring

Drift in data streams refers to the temporal evolution or evolving changes in the underlying statistical properties of the data (Agrahari and Singh, 2022). Detecting drift involves employing a methodology to determine and identify significant changes in data streams at a specific time point t. In industrial settings, detecting drift is a fundamental task that serves to indicate and diagnose the status of the ingested data, prompting appropriate actions in response to identified changes (Ditzler et al., 2015). Typically, drift is detected by quantifying the dissimilarity between the data probability distributions of two timestamps using the divergence metric. If the divergence metric exceeds a certain threshold ζ, it signifies a drift occurred in the data (Liu et al., 2013). This metric can be calculated as follows:

$$D_t = \delta(P \parallel Q). \tag{1}$$

The decision-making logic is defined as follows: If $D_t > \zeta$, a drift in the data is detected; otherwise, no detection occurs. Where D_t represents the divergence metric recorded at timestamp t between probability distributions P and Q, and δ is a function measuring this divergence.

In the context of nonstationary applications, determining a suitable threshold value ζ for detecting drift poses a significant challenge (Micevska et al., 2021). Industrial data streams are usually dynamic and complex, resulting in constantly changing statistical properties. This variability makes establishing a single, universal threshold impractical. Moreover, effectively detecting drift requires a subtle understanding of the specific industrial processes and the context of the application (Kammerer et al., 2019). As a result, recent research concludes that the drift detection threshold should not be fixed. Instead, it must be adaptive, adjusting dynamically to reflect the evolving nature of the system and its underlying processes (Liu et al., 2022).

To overcome these challenges in developing our proposed adaptive data quality scoring framework, we have integrated a dynamic method from our prior research, designed to efficiently detect changes in time-series data distributions (Bayram et al., 2023b). This method has been Incorporated into our proposed framework to monitor the distribution of incoming application data. Specifically, it actively observes the divergence values computed for each time window frame using a sliding window mechanism. The divergence value is calculated using the Jensen–Shannon divergence, defined as follows (Lin, 1991):

$$\mathrm{JSD}(P_{\mathrm{his}} \parallel P_{\mathrm{cur}}) := H\left(\frac{p_{\mathrm{his}} + p_{\mathrm{cur}}}{2}\right) - \frac{H(p_{\mathrm{his}}) + H(p_{\mathrm{cur}})}{2}, \tag{2}$$

where P_{his} and P_{cur} represent the probability distributions for the historical and current application data samples, respectively, and the function H is the Shannon's entropy given by:

$$H(p) = -\int p(\mathbf{Y}) \log p(\mathbf{Y}) d\mathbf{Y}. \tag{3}$$

Subsequently, the algorithm makes decisions on drift detection by performing hypothesis testing on the p-value of the observed divergence value, indicating the extremeness of the current change magnitude with respect to historical magnitudes. If the p-value falls below the defined significance threshold τ, a drift detection signal is triggered. An inherent advantage of this method lies in its avoidance of the requirement of a predefined threshold for drift magnitude. Furthermore, its dynamic nature ensures robustness and adaptability as the divergence distribution evolves with the accumulation of more data, making it a viable solution for the evolving nature of real-world industrial applications.

2.2. Data quality assurance

In the industrial applications of ML systems, the assurance of data quality is fundamental to developing high-performance decision-making support (Wang et al., 2023). In particular, rigorous data quality practices are essential to ensure that the data used for both training and inference are of optimal quality and represent the underlying processes in a timely manner (Fan and Geerts, 2022). Low-quality data in industrial processes that are driven by data analytics could be caused by machine errors, inconsistent sensor measurements, or abnormal patterns, among other interpretations. Therefore, data quality assurance is recognized as a critical factor that significantly impacts the cost-effectiveness and operational efficiency of industrial processes, addressing potential faults and improving overall performance.

Data quality assurance is a multi-dimensional concept that spans various attributes to collectively assess the validity of collected data (Pipino et al., 2002). Each dimension offers a unique perspective on the specific characteristics of the data. Numerous studies have compiled extensive lists, some identifying up to 179 distinct quality dimensions (Wang and Strong, 1996). These dimensions are further grouped into *intrinsic*, *contextual*, *accessibility*, and *representational* categories (Priestley et al., 2023). The selection of data quality dimensions is not uniform, as there is no universally accepted definition. Rather, they are often defined based on the specific goals and requirements of a given task and application (Karkouch et al., 2016).

Based on the data context of our use case, we have identified and selected specific data quality dimensions that are highly relevant and applicable to the characteristics of our industrial application. We summarize the definitions of data quality dimensions as follows:

1. **Accuracy**: This dimension evaluates how well the recorded data aligns with the actual values it is meant to represent. It evaluates precision and correctness, which are essential for detecting anomalies that could lead to product defects.
2. **Completeness**: This dimension assesses the thoroughness of the observed data by checking for missing values that are collected from the data source. Ensuring that all relevant sensor data is captured and no sensor failures occur.
3. **Consistency**: This dimension examines whether the observed values conform to the integrity constraints of the domain, ensuring that collected data records fall within the expected value ranges. This contributes to effective product quality monitoring and prevents errors due to incorrect sensor readings.
4. **Timeliness**: Describing the relevance of data for specific tasks, timeliness assesses whether the observed data is current. In dynamic industrial settings, where ongoing tasks often demand up-to-date information, ensuring the currency of collected data becomes of high importance. This ensures that data is still representing the system and can be trusted for the task.
5. **Skewness**: This dimension goes beyond traditional measures, computing the distribution deviation of observed data from a reference distribution. Addressing skewness is important for optimized performance, especially for ML systems, as skewed data distributions can affect model accuracy and generalization. It verifies that no unexpected changes or unusual patterns exist.

Observing the defined data quality dimensions, a key point of consideration emerges within the context of the ML systems: a subset of these dimensions, including timeliness and skewness, exhibit *dynamic characteristics*. This implies that the quality aspects of these dimensions may vary according to the prevailing conditions. To illustrate, for the timeliness dimension, what is deemed timely and well-fitted in the present may not remain true in the future. Similarly, the skewness of the data may vary with different seasons, introducing variables such as drift and seasonality. Therefore, data demonstrating drift during a specific season, resulting in a low skewness score for the current season, might conversely indicate a high score in a different season. On the other hand, certain dimensions, such as accuracy, completeness, and consistency, exhibit *constant characteristics*, maintaining their relevance irrespective of changing conditions. These constant dimensions focus on the precision and validity of data representation rather than relying on transient factors and settings within the problem domain. Therefore, the integration of the ML-based scoring framework with an adaptation

methodology specifically addresses the adaptive nature of data quality in dynamic dimensions within data-centric ML applications. This integration specifically addresses the adaptive nature of data quality in data-centric ML applications.

3. Related work

The pivotal role of data quality assessment in ensuring the reliability and effectiveness of data-driven processes has been examined in diverse applications and domains, highlighting its growing recognition in the research community. In the field of healthcare, such as electronic health records (EHRs), researchers have explored methodologies to assess and improve the quality of patient data, acknowledging its critical impact on medical decision-making and patient care (Lewis et al., 2023). Within the financial sector, studies have focused on assessing the quality of financial data to maintain the integrity of analytical models and regulatory compliance (Karkošková, 2023; Hasan et al., 2020). Assessment of spatial data quality has been studied by introducing a comprehensive quality assessment framework for linear features of Volunteered Geographic Information (VGI) by integrating novel quality metrics with those commonly used through factor analysis (Wu et al., 2021). The common thread between these diverse applications is the recognition of data quality as a foundational element for robust and trustworthy results (McGilvray, 2021).

The use of IoT technologies in industries to gather and generate data from IoT sensors, often in real-time, requires a rigorous evaluation of data quality (Mansouri et al., 2023). Taleb et al. (2021) presented the Big Data Quality Management Framework (BDQMF) as an exhaustive strategy aimed at addressing data quality challenges inherent in large-scale data systems. The framework defines various dimensions of data quality and incorporates multiple components dedicated to managing, validating, and monitoring data quality. It includes a scoring mechanism to quantify different aspects of data quality. The authors explored issues related to data quality at both the individual cell instance and the schema levels within datasets. Another recent approach in this context is the big data quality assessment framework (BIGQA) (Fadlallah et al., 2023). The framework provides a declarative solution specifically designed for non-expert users, featuring reporting functionality to visualize outcomes or scores indicating the quality of input datasets. In a different approach, the Data Quality Anomaly Detection Framework focuses on anomaly detection (Widad et al., 2023), based on an extended isolation forest model. It introduces the *Quality Anomaly Score* metric to evaluate the degree of anomalousness in six dimensions of quality.

In addition to these studies, Chug et al. (2021) have developed a method to assess dataset quality using nine dimensions, yielding a *comprehensive score*, *report*, and *label*. The study introduced *data quality ingredients* as semantic indicators, identifying nine crucial aspects, including provenance, characteristics, uniformity, metadata coupling, missing cells, duplicate rows, data skewness, inconsistency ratio in categorical columns, and attribute correlation. These ingredients contribute to the calculation of the final score. In a previous study, Ardagna et al. (2018) proposed a method to enhance computational efficiency by focusing on a specific data subset, reducing both time and resource requirements using parallelization. To convey the reliability of data quality values, they introduced a *confidence* metric tied to the considered data volume and influenced by time constraints and computational resources. The confidence metric serves as an indicator of the trustworthiness of the data in relation to the quality dimension of the data tested.

Another recent research Byabazaire et al. (2022) has explored *trust* metrics as a means of assessing real-time data quality in IoT deployments. The proposed framework allows end-users to tailor trust metrics, offering visibility into data quality throughout the big data model. Building on this trust metric for real-time data quality assessment in IoT deployments, another paper (Byabazaire et al., 2023) extends the previous approach to data quality assessment. This study introduces a solution that employs data fusion strategies, specifically Adaptive Weighted Fusion, Kalman's Fusion, and Naïve Fusion, to derive a unified quality score. The practical experiments aim to evaluate the computational efficiency of these different fusion methods.

The analysis of the existing literature highlights the widespread use of traditional methods in evaluating data quality, revealing a lack of transformative frameworks and significant breakthroughs. Furthermore, the majority of studies tend to assess complete datasets, which could pose challenges to ML systems. A more effective approach lies in the real-time scoring of the data streams, which offers a balance between efficiency and granularity that better suits the dynamic nature of ML systems. This aligns with the inherent reliance of ML systems on individual data records, as scoring at the record level allows for a more granular assessment of data quality. In addition, the significant costs associated with traditional scoring methods require alternative approaches. Therefore, several recent studies have explored ML-based techniques as a promising solution to mitigate these drawbacks. One such method is presented by Widad et al. (2023), which utilizes an intelligent anomaly detection model to score data quality anomalies using an unsupervised ML model. The data quality anomaly scoring framework detects anomalies in multiple quality dimensions and quantifies the extent of deviation from established quality standards. In another research, our previous framework Data Quality Scoring Operations (DQSOps) (Bayram et al., 2023a), based on DevOps principles, specifically continuous integration/continuous delivery (CI/CD), enables efficient real-time data quality evaluation in industrial applications. The framework scores several data quality dimensions and demonstrated significantly faster processing rates compared to traditional scoring processes. Although these ML-based methods effectively score the quality of data streams in a production environment, they lack an adaptive nature for data quality assessment, which we address in the current research.

4. The adaptive data quality scoring operations framework

The proposed adaptive data quality scoring framework employs an ML-based approach to score the data quality in industrial applications. The primary objective of the framework is to label the incoming data windows with a score that quantifies the quality of the data window based on various pre-defined data quality dimensions. This innovative framework has been designed to address the intrinsic limitations of the non-adaptive static data quality scoring framework, especially in industrial contexts characterized by dynamic data environments. Specifically, the novel adaptive framework effectively handles the dynamic aspects of data quality by re-assessing the data quality scores based on the prevalent conditions observed in the application system. This adaptation mechanism, which is based on the drift detection method, allows the dynamic data quality dimensions to conform to the evolving characteristics of the underlying data and ML processes. Furthermore, the drift detection method eliminates the need to define a fixed window size w to retrain the ML model, since the retraining signal is activated only when a drift is detected. The workflow of the proposed adaptive data quality scoring framework is illustrated in Fig. 2.

First, we provide an overview of the underlying ML-based scoring framework to establish a foundation and enhance understanding. This overview summarizes the framework's core components, specifically highlighting its implementation of MLOps practices to manage the ML regressor for data quality scoring. This context is crucial to understanding the operational environment of the proposed framework. Subsequently, we dissect the two integral phases to achieve the proposed framework: development and deployment phases. The development phase focuses on initializing the framework and generating the necessary artifacts. These artifacts are then streamlined in the deployment phase, enabling dynamic scoring of incoming data windows based on their quality.

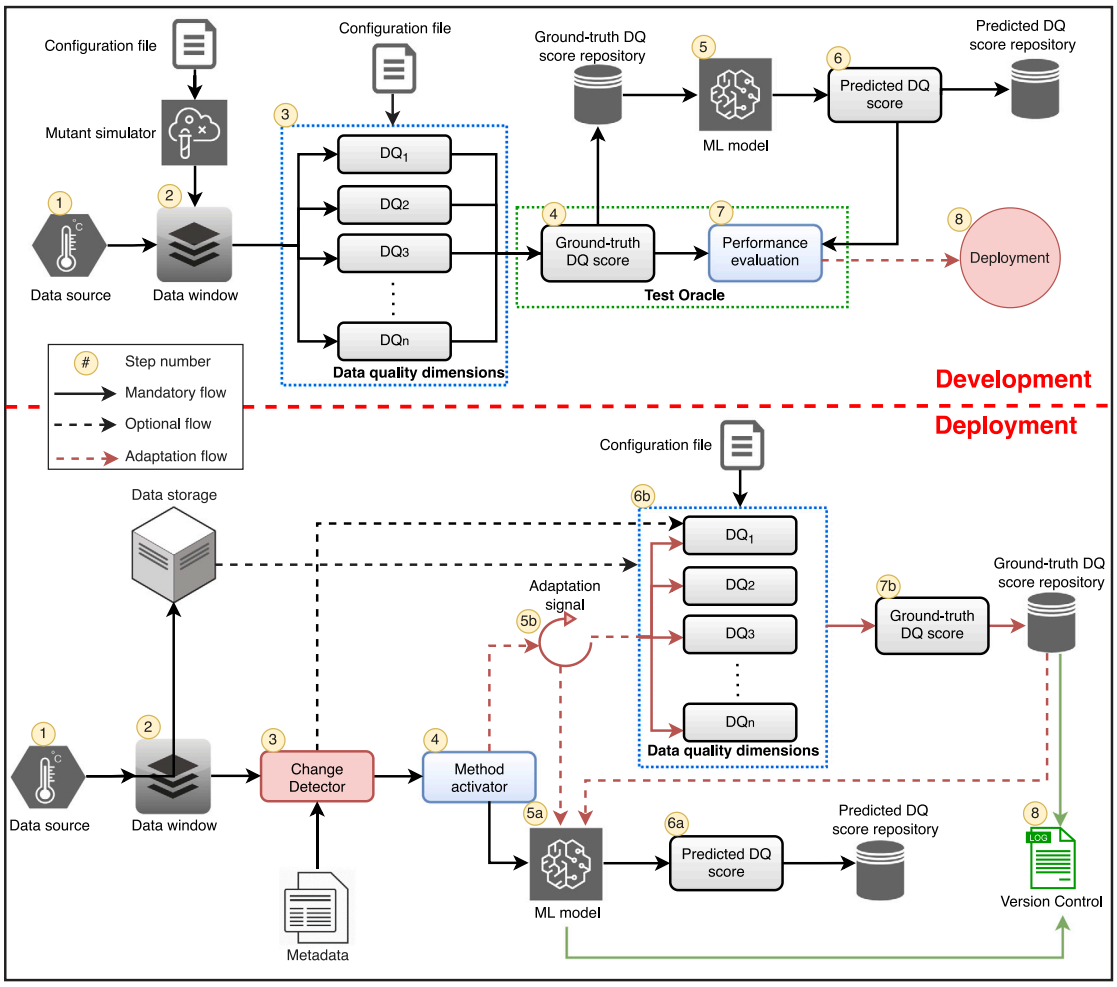


Fig. 2. The adaptive data quality scoring operations framework.

4.1. An overview of ML-based data quality scoring

The ML-based data quality scoring approach aims to determine the data quality of the collected data window to generate a unified score using an ML model (Evans, 2006). This score reflects the overall adequacy of the data windows, considering multiple dimensions of data quality. Specifically, the ML-based scoring framework is designed to address the complexity of scoring data quality in data-driven applications (Singh and Suri, 2015). The main innovation in the framework is streamlining the scoring process by using an ML predictor instead of traditional standard scoring methods following MLOps principles, incorporating continuous monitoring and validation practices. This integration provides significant speedup rates while maintaining high predictive performance levels. Additionally, the framework's runtime remains unaffected by the number of quality dimensions, offering practical scalability in real-world applications with high sampling rates.

To accomplish ML-based scoring operations, the framework workflow begins by initializing the ML predictor in a warm-start mode. In particular, the ML predictor is initiated using training data that contain ground-truth quality scores obtained by the standard-based approach. The model's prediction accuracy is monitored using a *test oracle* until it reaches a predetermined threshold T based on a performance metric. Once this threshold is met, the ML predictor is eventually deployed in the real-world problem. Meta-information files, such as the anomaly detection model and reference data distribution, are also prepared to calculate certain data quality dimension scores. Furthermore, as part of this phase, a mutant simulator is integrated to improve the model's learning by introducing a variety of data quality issues that may arise in practical situations, thus accelerating the learning process.

Once the framework is deployed in a production environment, the method activator becomes a key component in managing the pipeline flow. It employs predetermined criteria to select the appropriate approach to data quality scoring. Specifically, the activator chooses between the ML-based approach and the standard-based approach based on the collected data window and chunk size. The activator repeatedly executes the ML model to obtain data quality scores until the chunk size reaches a preconfigured threshold β. At this point, an evaluation is initiated using the standard-based approach to ensure continuous monitoring of the ML model's accuracy. This evaluation employs a specified test oracle to compare predicted quality scores with ground-truth scores, and if the model's performance falls below a pre-defined tolerance level, a retrain signal is activated.

The process of finding the consolidated data quality score involves aggregating the calculated values of the data quality dimensions. Traditional methods like the arithmetic mean may not be suitable due to their sensitivity to variable scales. To address this, the quality scores are standardized using z-scores, ensuring uniform integration of different data quality metrics (Heinrich et al., 2018). This standardization involves calculating the z-score for each element in the quality score matrix. Principal Component Analysis (PCA) is then employed, following widely used techniques in the literature (Teh et al., 2020).

PCA condenses the information from multiple dimensions into a single interpretable score, simplifying the analysis and facilitating easier comparison of data quality across different data windows or even datasets. Specifically, the first principal component, which indicates

the direction of maximum variance, is used as a score that signifies the overall quality of the data windows. This approach allows us to summarize the essential information from multiple dimensions into a single comprehensive score. This ensures that the overall quality assessment is more intuitive and easily understandable, facilitating better interpretation. Moreover, using a single comprehensive score rather than multiple dimensions makes the assessment process simpler and more straightforward for decision-making in our industrial context.

The quality of each time-series data is evaluated across several key dimensions, each representing a distinct aspect of data quality. Let TS_i represent a time series, where i denotes the index of the time series. The quality of each time series TS_i is evaluated across several key dimensions $DQ_1, DQ_2, \ldots, DQ_n$, each representing a specific aspect of data quality. The evaluation process yields a corresponding data quality score DQS_i for the time series TS_i, which is computed as follows:

$$TS_i \rightarrow \text{PCA}(DQ_1, \ldots, DQ_n) = DQS_i.$$

For the calculation of the individual data quality scores, the ground-truth labels for each dimension are determined using commonly used criteria in the literature (Taleb et al., 2021; Bayram et al., 2023a).:

1. **Accuracy Score:** Calculated as the proportion of anomalous data in the window: $Accuracy = \frac{NAV}{N}$, where NAV is total anomalous values, and N is the data window size.
2. **Completeness Score:** Quantifies missing values: $Completeness = \frac{NNV}{N}$, where NNV is the number of missing values (NA or NULL), and N is the window size.
3. **Consistency Score:** Calculated based on defined integrity constraints for data values: $Consistency = \frac{NCV}{N}$, where NCV is the number of consistent values, and N is the window size.
4. **Timeliness Score:** Involves a goodness-of-fit test, particularly a two-sample Kolmogorov–Smirnov test, comparing current data with an expected distribution. The Kolmogorov–Smirnov test statistic is computed as $KS = \max_{1 \le i \le N} |\hat{F}_1(Z_i) - \hat{F}_2(Z_i)|$, where Z is the combined sample of two independent random samples X and Y.
5. **Skewness Score:** Utilizes the Jensen–Shannon Divergence (JSD) value to measure dissimilarity between the distributions of current and historical data. JSD is calculated as $\text{JSD}(P \parallel Q) = H\left(\frac{P+Q}{2}\right) - \frac{H(P)+H(Q)}{2}$, where H denotes Shannon's entropy. JSD is chosen for its bounded nature in the interval $[0, 1]$ (Lionis et al., 2021).

4.2. Development phase

Similar to any supervised ML system that requires a warm start for effective initialization and optimal performance (Ash and Adams, 2020), the development phase of our proposed framework starts with the initiation of crucial system artifacts essential for streamlining the solution in production. This phase includes tasks such as system setup, parameter configuration, establishing the anomaly detector, defining data distribution building parameters, managing metadata to calculate data quality dimension scores, ML model development, and preparing all necessary components for subsequent deployment. Additionally, certain meta-information helps reduce the number of calculations in the deployment phase. Table 1 summarizes the overall artifacts produced in the development phase. The overall workflow of this phase is illustrated in Fig. 2. The process begins with data collection from the data source **(Step 1)**, which is then segmented into data windows **(Step 2)**. These data windows are processed through various data quality dimensions **(Step 3)**, and the necessary auxiliary information for calculating these dimensions, such as integrity constraints for consistency score and file paths, is loaded from a configuration file. This information is used to calculate the individual ground-truth quality scores **(Step 4)**.

The core outcome of this phase is the creation of the ML predictor for data quality **(Step 5)**. This predictor is trained on historical data labeled with ground-truth quality scores **(Step 6)**. These scores are calculated using the standard approach detailed in Section 4.1. To enhance the predictor's learning capabilities and accelerate training, a *data mutant simulator* is employed. This component introduces potential data quality issues that may occur in real-world scenarios, enhancing the predictor's ability to learn data quality issues and potentially reducing the amount of real-world data required. The mutation parameters are stored in configuration files, which include settings like the percentage of faults to be simulated. Continuous performance monitoring is implemented using a *test oracle* to systematically evaluate the performance of the ML predictor **(Step 7)**, with a predefined threshold for a selected performance metric. Consistent achievement of this threshold indicates sufficient learning for deployment to the production environment **(Step 8)**, marking the completion of the development phase.

Table 1
Artifacts produced in the development phase.

Artifact	Description
ML predictor	The ML model that is used to predict quality score.
Divergence values	Used for the dynamic drift detection method to detect changes.
Anomaly detector	Utilized in the calculation of the accuracy score.
Data distribution	Represents the historical PDF used to calculate skewness score.
Historical samples	Used in the timeliness score calculation.

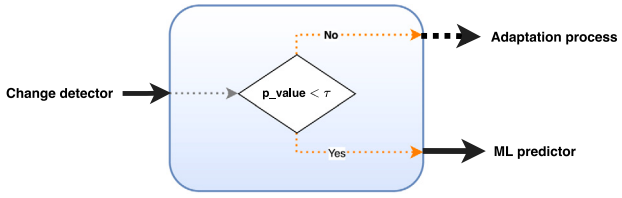


Fig. 3. Method activator component flowchart.

4.3. Deployment phase

Moving from the development environment to the production environment, the deployment phase of our adaptive framework involves incorporating the developed and related components of the ML predictor and integrating them into the operational system. To manage the ML system in this phase, the principles of MLOps are followed, including CI/CD. The adoption of MLOps deployment strategies results in a systematic and efficient process for deploying and maintaining the ML model in a live operational context (Kreuzberger et al., 2023). The MLOps ecosystem encompasses a broad set of practices, tools, and techniques designed to automate the ML life cycle. This includes version control, automated testing, continuous monitoring, and continuous model updates, which makes it valid for industrial applications with minimal intervention (Kumara et al., 2023).

As illustrated in Fig. 2, the real-time data collected from the source **(Step 1)** is segmented into data windows **(Step 2)** during the deployment phase. Subsequently, a change detector is applied to the data window to assess the occurrence of drift **(Step 3)**. The change detector performs hypothesis testing to determine the significance of the drift magnitude, as explained in detail in Section 2.1. The change detector leverages meta-information collected during the development phase, such as historical PDF for data and divergence values, both of which are used in making the calculations in this step. The result is then forwarded to the method activator component, which in turn makes decisions about which flow to proceed with **(Step 4)**. The flow chart diagram of the activator component of the method is depicted in Fig. 3.

In the absence of detected drift, the method activator component continues using the current ML model **(Step 5a)** without initiating

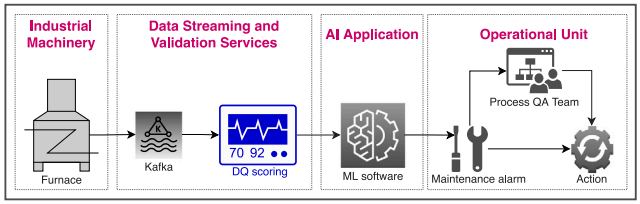


Fig. 4. Integration of data quality scoring in the holistic industrial use case application.

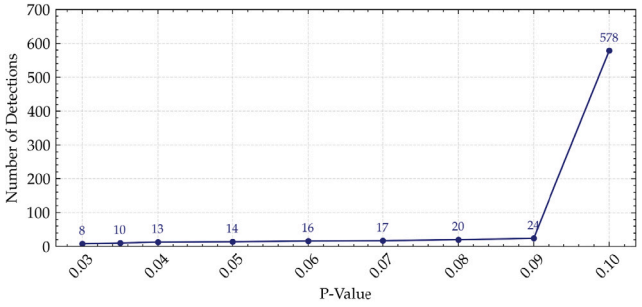


Fig. 5. Changes detected at different p-values.

adaptation to make the DQ scoring predictions **(Step 6a)**. However, in the case of drift detection, the activator component of the method initiates a retraining signal **(Step 5b)**. This signal triggers the adaptation process flow, which simulates the re-scoring of the historical data based on the prevailing conditions and updates the scores of the dynamic data quality dimensions, and scoring the newly collected data **(Step 6b)**. This mechanism ensures that the previous ground-truth labels are properly updated to reflect the changes in the dynamic data quality dimensions. Subsequently, the ML model is retrained from scratch using both the updated development-time training data and the newly scored data points **(Step 7b)**. This approach ensures that the model fully integrates the updated ground-truth labels for historical data, reflecting the changes in dynamic data quality dimensions. Additionally, a version control system is employed to manage and track updates to both the data quality scores and the ML model **(Step 8)**, ensuring that frequent updates are systematically documented and managed.

5. Experimental results

To evaluate our proposed framework, practical experiments were carried out in collaboration with Uddeholms AB, a leading steel manufacturer.[1] These experiments were carried out to assess the efficiency of the framework from various perspectives, including the accuracy of predictive performance, execution time, and resource consumption over time. Furthermore, we conducted a comparative analysis between our adaptive approach, the static data scoring approach, and the standard scoring approach. This analysis is presented to provide detailed insight into the merits and limitations of each approach. The following subsection presents a detailed description of the industrial use case and implementation, followed by a thorough analysis and discussion of the experimental results.

5.1. Implementation and use case details

Our adaptive approach is designed to be streamlined into a broader software system specifically developed to assist decision-making in industrial processes. In this set of experiments, the proposed framework was implemented and evaluated in the industrial application use case of the Electroslag Remelting (ESR) vacuum pumping process at the Uddeholm steel manufacturer in Sweden. The overall AI-driven application aims to sustain the production of high-quality steel by monitoring the pressure values. A representation view of our studied application is shown in Fig. 4.

Starting with industrial machinery, the figure shows the key ingredients of the application, including the furnace where a sensor collects pressure data within the vacuum chamber. Each time the vacuum pump is activated, which can take up to 20 min, pressure values are continuously monitored in this use case. The sensor records values every millisecond and through the Apache Kafka streaming platform,[2] data windows are transmitted every second for real-time analysis. Subsequently, each data window proceeds to data streaming and validation services, passing through a data quality scoring framework. Subsequently, the scoring framework validates the collected data, assesses its quality, and generates a score. Following this, in the AI-driven decision-making process, alarms are triggered for improper pump events. These alerts are communicated to the maintenance team of the operational unit, which takes appropriate actions, such as stopping the pump event to eliminate its costs. The primary purpose of the application is to achieve a gradual decrease in pressure during appropriate pump events, reaching the desired minimum value within the allocated time while promptly identifying and addressing improper pump events to prevent interruptions and uphold optimal furnace operation.

In terms of implementation details, our proposed framework was built with the Python programming language, taking advantage of its multipurpose capabilities and wide range of libraries. The implementation also incorporates the YAML format for defining configuration files. These YAML-formatted configuration files allow for the clear and organized specification of different parameters and settings, ultimately enabling the flexibility and customization of the framework to meet specific requirements of the use case. For the ML models, we employ the widely-used extreme gradient boosting (XGBoost) model (Chen and Guestrin, 2016). This decision tree-based ensemble technique has proven effective across various applications, demonstrating its suitability for industrial contexts and its capability to provide strong and easily understandable results (Kiangala and Wang, 2021).

5.2. Drift detection sensitivity analysis

In a real-world production environment, the effectiveness of the adaptive data quality scoring approach is highly dependent on the change detection component to initiate the adaptation signal, a mechanism detailed in Section 4.3. A sensitivity analysis of the drift detection mechanism in the approach was conducted to understand the behavior of the change detector. This analysis involved varying the significance thresholds represented by the p-values. The results, visually presented in Fig. 5, illustrate the number of changes detected at different p-values, with the values tested including $\tau = [0.03, 0.035, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1]$.

The analysis reveals a gradual increase in the number of detections from a significance level τ of 0.03 to 0.09, ranging from 8 detections to 24, respectively, demonstrating a steady response to subtle variations. However, a significant spike is observed at a p-value of 0.1, reaching 578 detections, indicating a potential higher sensitivity that could lead to false alarms. Therefore, to maintain a diverse set of detection numbers in our experiments, we selected p-values of 0.03, 0.04, 0.06, 0.08, and 0.09, excluding 0.1 due to its observed high sensitivity.

5.3. Performance analysis of DQ scoring predictions

The predictive performance of our data quality scoring ML model in the adaptive framework through a series of experiments using the

[1] https://www.uddeholm.com/en/.
[2] https://kafka.apache.org/.

defined levels of significance threshold τ. The evaluation involves calculating the errors of the XGBoost regressor results, measured by two metrics: Mean Absolute Error (MAE) and R-squared (R2), which are given by:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \tag{4}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{5}$$

Where n is the number of data points, y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\bar{y}$ is the mean of the actual values.

The predictive performance of the framework was evaluated by examining the observed errors over time, specifically MAE, presented in Fig. 6, and R2 in Fig. 7. Each subfigure displays the error metric along with the corresponding detected drifts for each significance level τ, annotated with shaded areas representing the highest and lowest performances among the remaining levels. The results show that drifts are detected; hence, adaptation is executed more frequently in the initial stages of deployment than in later stages. This observation suggests that the drift detection mechanism becomes more robust over time, resulting in fewer false alarms as more divergence values are collected.

For prediction errors, the results show that after the execution of adaptation mechanisms prompted by drift detection, the predictive performance often improves. This improvement is particularly reflected in the decline in MAE and an increase in R^2 metrics after adaptation. Specifically, at the end of the experimental duration, the MAE metric values are 0.136 for experiments with $\tau = 0.03$ and 0.112 for experiments with $\tau = 0.09$. The values for the R^2 metric are 0.949 for experiments with $\tau = 0.03$ and 0.978 for experiments with $\tau = 0.09$. These variations in results are attributed to the fact that a lower τ leads to a reduced sensitivity, resulting in fewer triggered adaptations that may not be sufficient to update the predictor, while a higher τ leads to more frequent adaptations.

We extend the analysis to evaluate the performance of the non-adaptive static approach, a framework dependent on a fixed window size w for adaptation. Unlike the adaptive approach, where adaptation is determined by significance level τ, the static approach triggers adaptation if the window size has been reached and the ML performance falls below a defined level. The experiments conducted with the static approach involve varying window sizes for adaptation triggers, with four different window sizes w: 25, 50, 100, and 200. Smaller window sizes represent a more frequent assessment for adaptations, while larger window sizes offer a wider temporal range. The corresponding MAE and R^2 metrics for the static approach are presented in Figs. 8 and 9, respectively. Each subfigure within these figures illustrates the cumulative error metrics over time for different window sizes. As with our adaptive approach, the shaded area represents the range of performance for static approaches with other window sizes.

The results show that smaller window sizes show superior performance in both the MAE and R^2 metrics. Specifically, for a window size of 25, the MAE at the end of the experimental period is 0.11, while it is 0.257 for a window size of 200. Furthermore, the metric R^2 for a window size of 25 is 0.968, while it is 0.901 for a window size of 200. Similarly to our adaptive approach, there is a noticeable improvement following the adaptation process. This improvement is particularly evident in the plot for window size 200, as depicted in Figs. 8(d) and 9(d). Specifically, these figures illustrate a significant performance drop before reaching the window size of 200, indicating the need to update the predictor. In contrast, the window size of 25 follows a more frequent adaptation pattern, resulting in a faster recovery from performance drops, as shown in Figs. 8(a) and 9(a).

When comparing our adaptive approach with the static approach, we can observe that both frameworks follow similar performance at the end of the experimental period. For the adaptive approach, optimal performance is observed at a significance threshold of 0.03, which aligns the performance of the static approach with a window size of 25. However, as we deviate from the optimal adaptation parameter of each methodology, performance starts to degrade gradually. Additionally, during the initial phases of the experiments, the performance of both adaptive and static approaches is more volatile, which can be explained by the models' sensitivity to changes in input, but as time progresses, the predictor increasingly matures and its performance becomes more consistent and stable. Meanwhile, the static approach displays a wider range of performance than the adaptive, reflecting the cumulative effect of errors across various window sizes.

5.4. Time required analysis

The time required for various configurations was analyzed to understand their computational efficiency in the context of industrial tasks. Specifically, a comparison was performed between different scoring methodologies, including the adaptive, static, and the standard scoring approaches, to assess the time overhead in managing the data streams. The results of the time required to process the scoring task for the different approaches are summarized in Fig. 10.

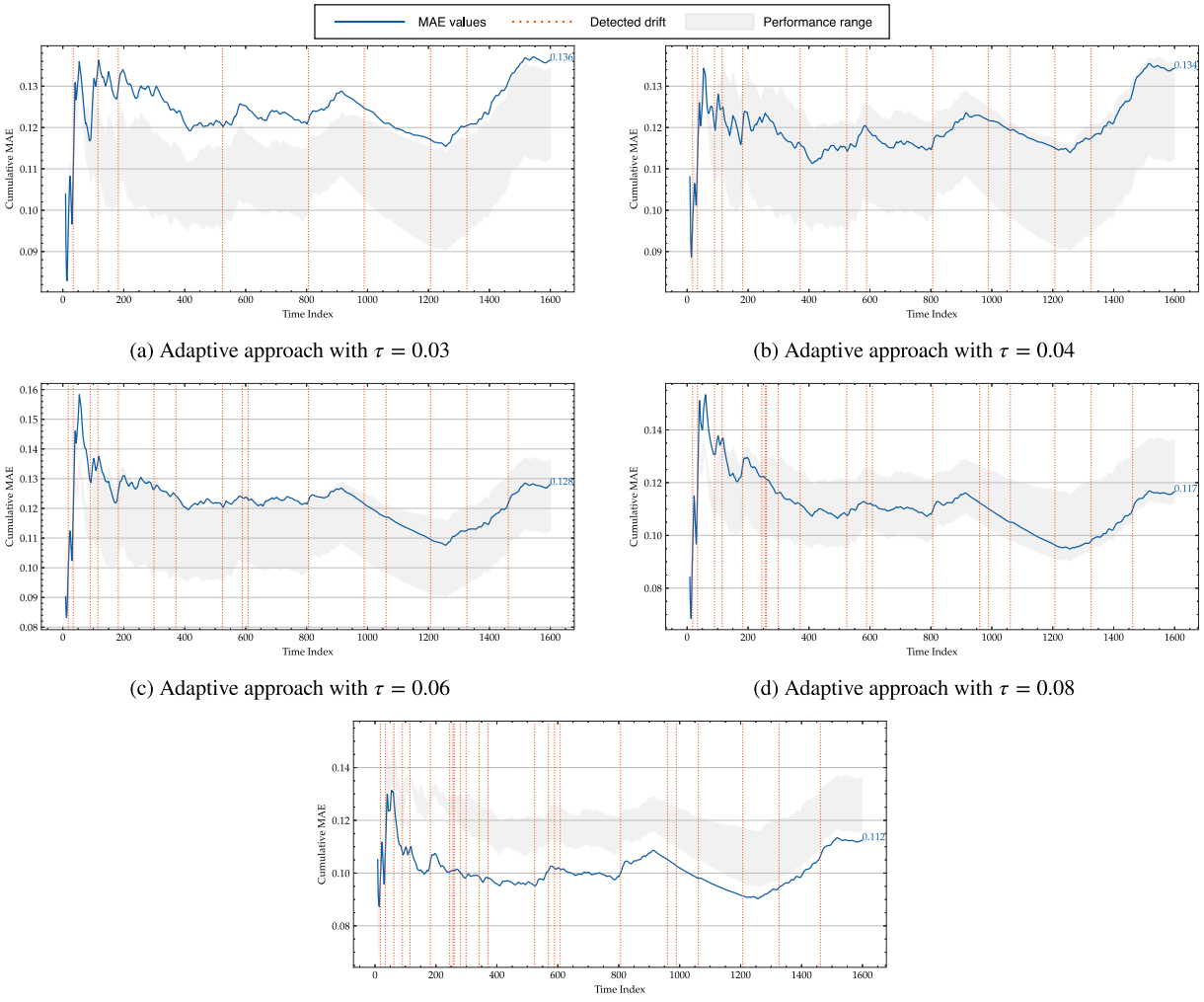
Of all the approaches tested, we can see that the standard scoring methodology was the most time-consuming during the experimental period, requiring a total processing time of 850.45 s. However, the static approach with a window size of 200 achieved the shortest processing time among all methods due to the lower frequency of adaptation triggers associated with larger window sizes. We can also observe that both the standard scoring and static approaches exhibited linear trends, with processing times accelerating rapidly over time. In contrast, the adaptive approach showed a more conservative trend, with processing times increasing at a slower pace. The discrepancy is the result of the periodic adaptation mechanism inherent in the static approach, whereas the adaptive approach only adapts when drift is detected, leading to a more regulated processing speed.

Furthermore, the static approach demonstrated higher sensitivity to variations in the adaptation parameter, particularly the window size w, leading to increased dispersion in processing times. In particular, as we move from a larger w of 200 to smaller sizes like 25, there is a substantial increase in processing time, from 62.15 s to 515.30 s. This sensitivity is also observed to a lesser extent in the adaptive approach. As indicated by the results, the adaptive approach with $\tau = 0.03$ required a processing time of 102.05 s and reached 203.91 s for $\tau = 0.09$.

Overall, the time analysis reveals significant improvements in both processing efficiency and scalability with the adaptive approach. Specifically, the adaptive approach with $\tau = 0.03$ achieved a processing time of 102.05 s an 88% reduction compared to the 850.45 s required by the standard scoring approach, representing a speedup factor of approximately 8.3x. In terms of scalability, Fig. 10 illustrates that the processing time for the standard approach grows linearly with data volume, while our adaptive approach shows a more restrained growth rate. These trends suggest that the adaptive approach would maintain its efficiency advantage even as data volume scales.

5.5. Analysis of dynamic data quality dimension scores

In this section, we analyze the evolution of dynamic data quality dimensions, specifically timeliness and skewness, under various drift occurrences. The analysis focuses on ten data points to highlight the dynamic nature of these data quality dimensions over time. The results of the discrepancy between each data point's score before and after each drift occurrence (for $\tau = 0.03$ across 8 drift occurrences), indicating how much the score changed, are summarized in Figs. 11(a) and 11(b), respectively. The numbers presented show the difference in the skewness and timeliness scores, highlighting how adaptation affects the score of the respective data quality dimension.



(a) Adaptive approach with $\tau = 0.03$

(b) Adaptive approach with $\tau = 0.04$

(c) Adaptive approach with $\tau = 0.06$

(d) Adaptive approach with $\tau = 0.08$

(e) Adaptive approach with $\tau = 0.09$

Fig. 6. Cumulative MAE for our adaptive approach with varying levels of τ. Shaded area denotes performance range across other τ values.

The findings reveal a substantial difference in magnitude after each adaptation, up to 0.3 for skewness and 0.4 for timeliness. The most significant change occurs immediately after the first occurrence after production, indicating that the system is more sensitive to change during this period. These experiments indicate a significant scale of change in the data quality dimension scores, signifying the impact of evolving patterns of the underlying data distribution. Furthermore, this shows the important role of adaptation in rescaling the data based on the prevalent situation, ensuring that the data quality dimensions accurately reflect the evolving characteristics of the dataset as their scores evolve over time.

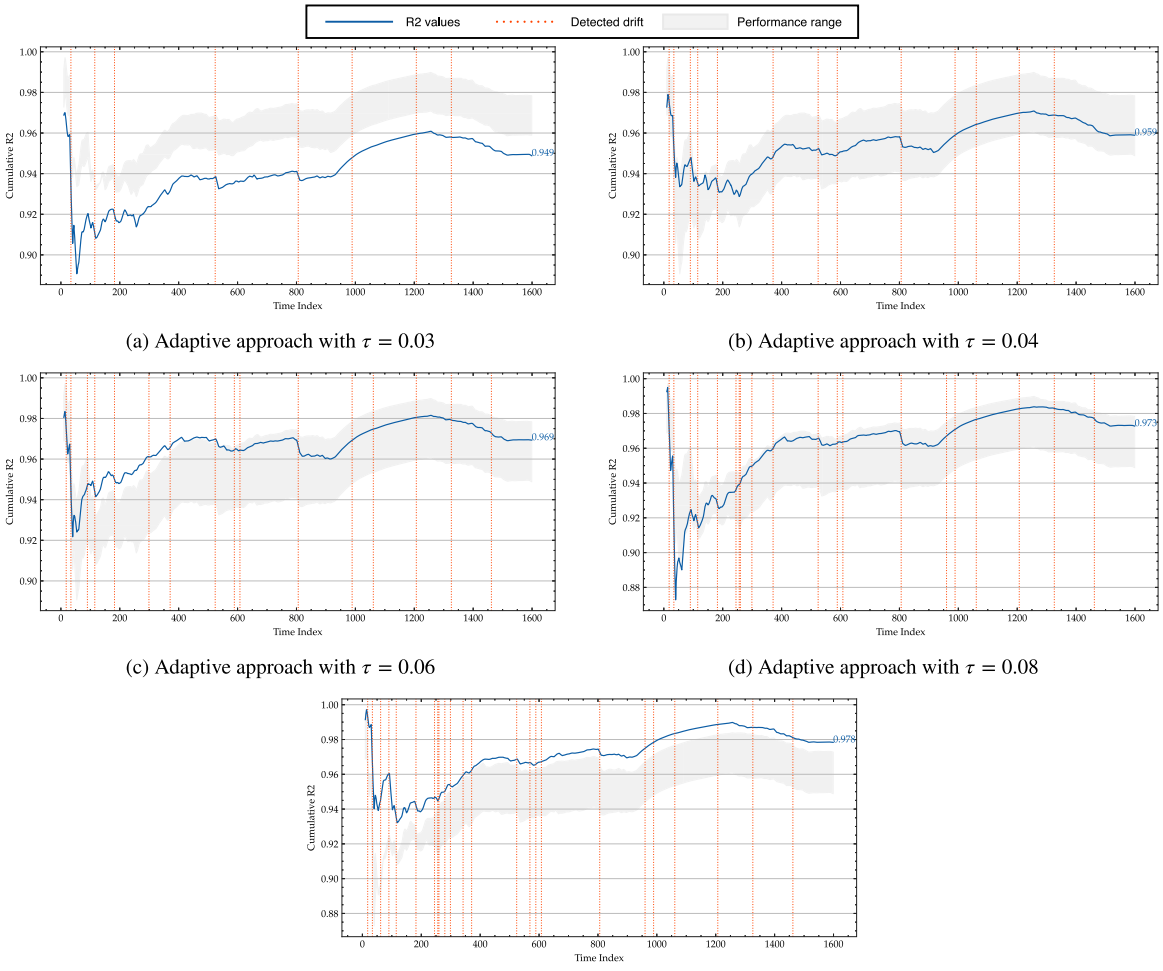
5.6. Resource consumption

The analysis of resource consumption explores the percentage of CPU and memory usage for each scoring approach, providing information on their respective resource demands in terms of computational and memory requirements. This analysis directly impacts operational costs and system efficiency, which are crucial factors in determining the most suitable approach to employ in production by industries. The percentage of CPU usage of the different approaches is summarized in Fig. 12, while memory usage is summarized in Fig. 13.

The results indicate that the adaptive approach consumes slightly more CPU compared to the static, while the memory consumption between the two approaches is very similar. This difference in resource consumption can be attributed to the dynamic change detector mechanism employed by the adaptive approach, which actively monitors and updates the ML predictor based on detected drifts in the data streams, resulting in slightly higher computational overheads. On the contrary, the standard approach showed the lowest CPU and memory usage across the board due to its lack of ML involvement, resulting in a more straightforward and less resource-intensive process. Moreover, the analysis also reveals that variations in parameters within each approach do not significantly impact the memory consumption percentage, and the difference in the boxes is negligible. However, the distinction is more apparent when considering CPU usage, primarily due to the ML operations involved. Additionally, the adaptive approach tends to exhibit wider ranges between the CPU and memory usage whiskers compared to the static approach. This characteristic arises from the detection mechanism, which occasionally triggers adaptations in the ML predictor. These adaptations lead to higher fluctuations in resource usage over time.

5.7. Operational insights and key takeaways

In this set of real-world industrial experiments, the behavior of data quality scoring approaches was analyzed in terms of prediction performance and processing time efficiency. In the following points, we analyze the main findings and lessons learned with an emphasis on practicality, aiming to offer valuable insights for industrial deployments:



(a) Adaptive approach with $\tau = 0.03$

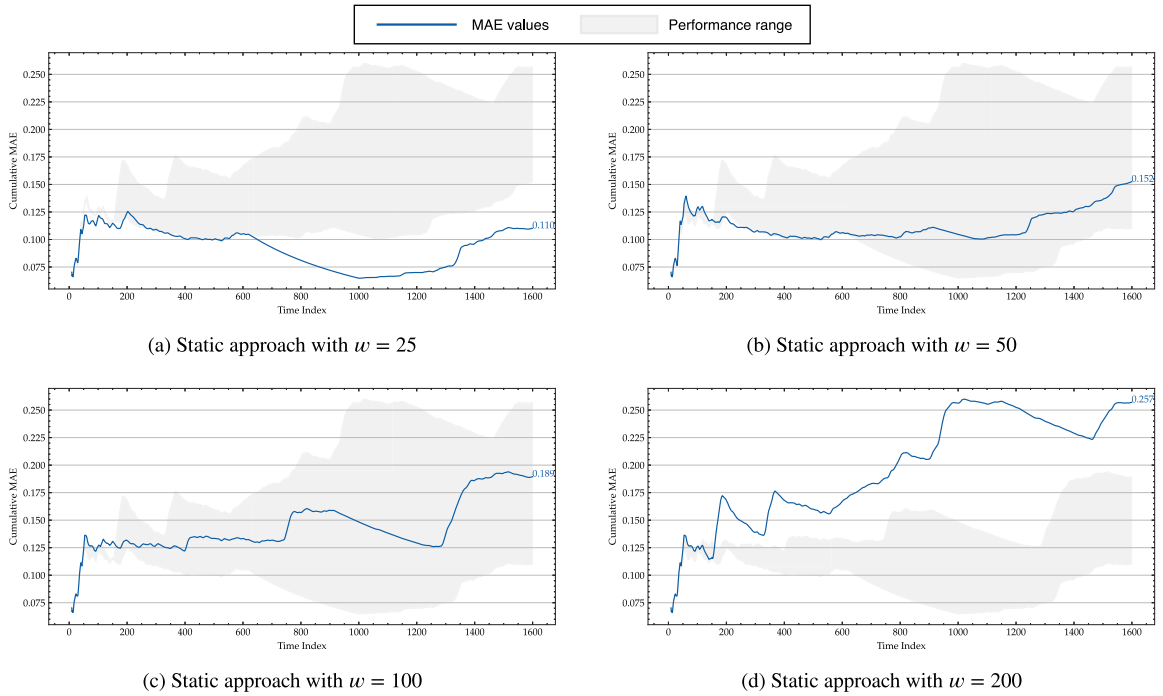
(b) Adaptive approach with $\tau = 0.04$

(c) Adaptive approach with $\tau = 0.06$

(d) Adaptive approach with $\tau = 0.08$

(e) Adaptive approach with $\tau = 0.09$

Fig. 7. Cumulative R2 for our adaptive approach with varying levels of τ. Shaded area denotes the performance range across τ values.



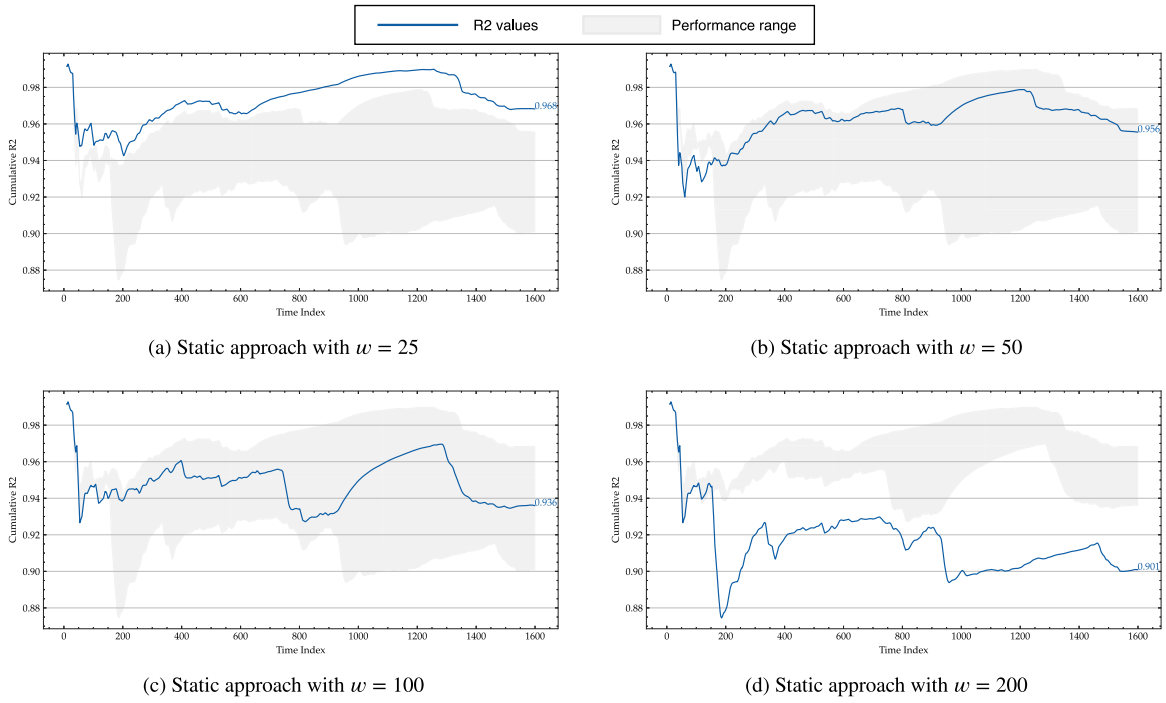
(a) Static approach with $w = 25$

(b) Static approach with $w = 50$

(c) Static approach with $w = 100$

(d) Static approach with $w = 200$

Fig. 8. Cumulative MAE for the static approach with varying window sizes w. Shaded areas denote performance range across different window sizes.



(a) Static approach with $w = 25$

(b) Static approach with $w = 50$

(c) Static approach with $w = 100$

(d) Static approach with $w = 200$

Fig. 9. Cumulative R2 for the static approach with varying window sizes. Shaded areas denote performance range across different window sizes.

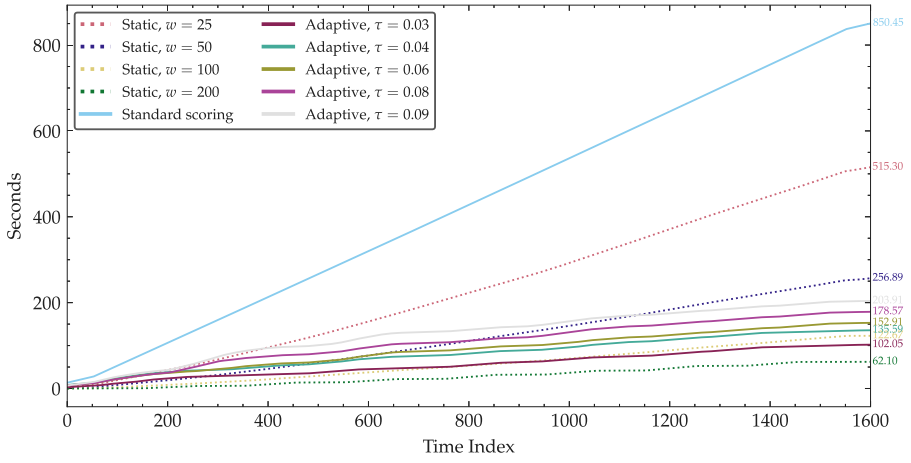
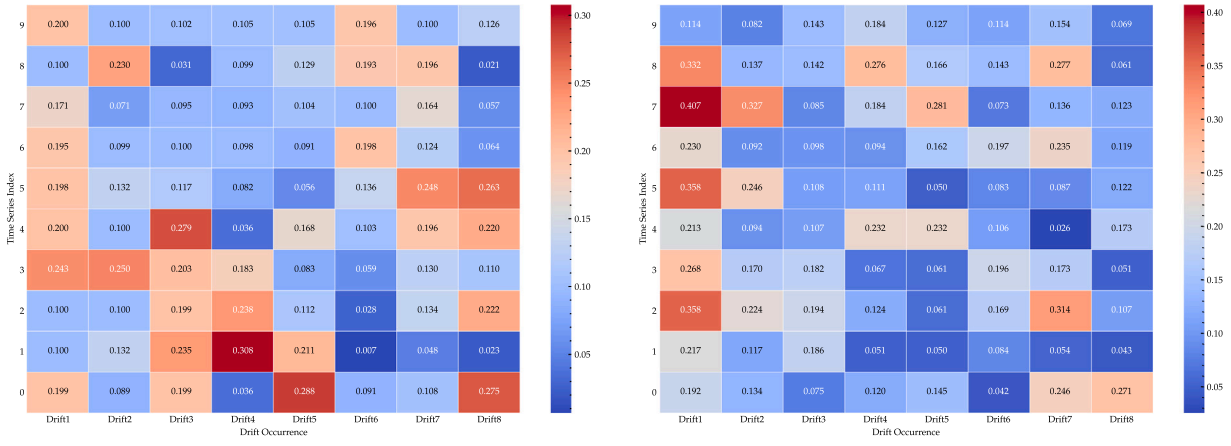


Fig. 10. Cumulative time elapsed (in seconds) over time for the different scoring methodologies.



(a) Change in skewness score after drift occurrences.

(b) Change in timeliness score after drift occurrences.

Fig. 11. Difference in skewness and timeliness scores before and after drift occurrences.

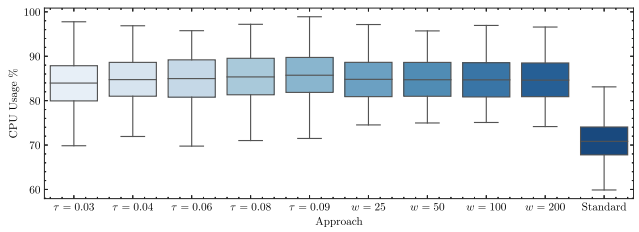


Fig. 12. Summary of CPU usage percentage for the different scoring approaches.

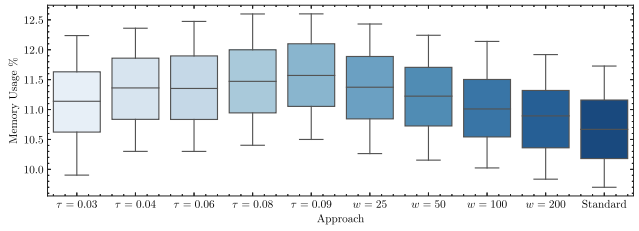


Fig. 13. Summary of memory usage percentage for the different scoring approaches.

1. **Improved Predictive Performance:** We found that both the adaptive and static approaches show improvements in predictive performance after executing adaptation mechanisms. This improvement is reflected in the decrease in MAE and the increase in R^2 metrics post-adaptation. For adaptation parameters, smaller window sizes w and lower significance thresholds τ resulted in higher performance improvements.
2. **Enhanced Processing Efficiency:** Regarding time efficiency, both frameworks demonstrated significant reductions in processing time compared to the standard data quality scoring approach. Comparing the two frameworks, we observed similar levels of predictive performance at the end of the experimental period using the optimal adaptation parameter. However, the adaptive approach achieved this performance with a more than two-fold improvement in processing time. This efficiency gain can be attributed to the fact that the adaptive approach triggers adaptations only when necessary, thus eliminating the costs associated with unnecessary adaptation processes. Moreover, both frameworks showed discernible trends in terms of acceleration and dispersion of processing time. The static approach showed a higher magnitude of acceleration in processing time. However, the adaptive approach displayed a more conservative trend, with processing times increasing at a steadier pace. This difference arises from the periodic adaptation mechanism inherent in the static approach, whereas the adaptive approach adapts only when drift is detected, resulting in a more regulated processing trendline.
3. **Sensitivity to Adaptation Parameters:** The static approach demonstrated a higher sensitivity to changes in the adaptation parameter, particularly the window size w, leading to a greater dispersion in processing times as this parameter varied. On the contrary, the adaptive approach showed less sensitivity to these changes, resulting in more consistent processing times over varying thresholds τ. As the system progresses over time, this trend implies potential cost savings, serving as an indicator of the efficiency gained by integrating a drift detector to control the adaptation processes.
4. **Minimal Impact on Resource Consumption:** The resource consumption analysis revealed that the adaptive approach utilizes slightly more CPU compared to the static approach, while the difference in memory usage between the two approaches is trivial. However, this additional CPU consumption is not significantly higher, implying that the differences in resource demands between the two approaches are relatively small. This indicates that the modifications executed in the adaptive approach do not add a significant burden on the data scoring system. Moreover, the standard approach was observed to be the least consuming in terms of CPU and memory resources, mainly because it does not involve ML components.
5. **Key Industry Implications:** In our industrial context, maintaining high performance and ensuring timely processing of accurate data is crucial for critical use cases, such as preventing costly improper pumping events. The data quality scoring, as a component of a broader data-driven AI system, is crucial for making sound decisions. The analysis showed that the adaptive approach, with its dynamic nature, ensures that data quality scores remain up-to-date, reflecting the most recent state of the data streams. By activating adaptations only when required, the adaptive approach maintains current data scores and saves costs by avoiding unnecessary processing overheads. This adaptability is particularly advantageous in dynamic industrial environments where data characteristics may change over time, ensuring that the information derived from data quality is recent and reliable. Therefore, this dual benefit of up-to-date scores and cost savings demonstrates the benefits of the adaptive approach over other frameworks.

6. Conclusion

Assuring the quality of the incoming data streams is essential to build data-driven AI systems. An approach to quality assessment is data quality scoring, which allows quantification and evaluation of data quality. In this paper, we introduce the adaptive data quality scoring framework, which is designed to address the challenges of scoring dynamic data quality dimensions in industrial processes. The adaptive scoring framework is especially valuable in scenarios where scores should align with the current conditions of the system. The primary novelty of our framework is the use of a dynamic change detector, which controls the adaptation of the ML scoring predictor based on detected drifts in the data streams. Our proposed framework actively updates the ML predictor only when necessary, rather than passively updating it periodically, thus minimizing unnecessary computational overhead. Our experimental results demonstrate that implementing our framework leads to substantial enhancements in processing time efficiency with negligible impact on resource consumption and overall improved performance within a real-world use case. These enhancements are particularly reflected in the high predictive performance and reduced processing time, making our framework a feasible solution for critical industrial applications. Moving forward, we will investigate the integration of our framework into broader data-driven AI systems, where its real-time data quality scores can inform decision-making industrial processes by leveraging these quality scores to enhance the training of ML models on high-quality data.

CRediT authorship contribution statement

Firas Bayram: Writing – original draft, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Bestoun S. Ahmed:** Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Erik Hallin:** Writing – review & editing, Validation, Software, Resources, Data curation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Bestoun S. Ahmed reports financial support was provided by Knowledge Foundation of Sweden (KKS). If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

This work has been funded by the Knowledge Foundation of Sweden (KKS) through the Synergy Project AIDA - A Holistic AI driven Networking and Processing Framework for Industrial IoT (Rek:20200067).

References

Agrahari, S., Singh, A.K., 2022. Concept drift detection in data stream mining: A literature review. J. King Saud Univ. Comput. Inf. Sci. 34 (10), 9523–9540.

Ardagna, D., Cappiello, C., Samá, W., Vitali, M., 2018. Context-aware data quality assessment for big data. Future Gener. Comput. Syst. 89, 548–562.

Ash, J., Adams, R.P., 2020. On warm-starting neural network training. Adv. Neural Inf. Process. Syst. 33, 3884–3894.

Batini, C., Cappiello, C., Francalanci, C., Maurino, A., 2009. Methodologies for data quality assessment and improvement. ACM Comput. Surv. (CSUR) 41 (3), 1–52.

Bayram, F., Ahmed, B.S., Hallin, E., Engman, A., 2023a. DQSOps: Data quality scoring operations framework for data-driven applications. In: Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering. pp. 32–41.

Bayram, F., Aupke, P., Ahmed, B.S., Kassler, A., Theocharis, A., Forsman, J., 2023b. DA-LSTM: A dynamic drift-adaptive learning framework for interval load forecasting with LSTM networks. Eng. Appl. Artif. Intell. 123, 106480.

Budach, L., Feuerpfeil, M., Ihde, N., Nathansen, A., Noack, N., Patzlaff, H., Naumann, F., Harmouch, H., 2022. The effects of data quality on machine learning performance. arXiv preprint arXiv:2207.14529.

Byabazaire, J., O'Hare, G.M., Collier, R., Delaney, D., 2023. Iot data quality assessment framework using adaptive weighted estimation fusion. Sensors 23 (13), 5993.

Byabazaire, J., O'Hare, G.M., Delaney, D.T., 2022. End-to-end data quality assessment using trust for data shared IoT deployments. IEEE Sens. J. 22 (20), 19995–20009.

Chen, H., Chen, J., Ding, J., 2021. Data evaluation and enhancement for quality improvement of machine learning. IEEE Trans. Reliab. 70 (2), 831–847.

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining. pp. 785–794.

Chen, H., Hailey, D., Wang, N., Yu, P., 2014. A review of data quality assessment methods for public health information systems. Int. J. Environ. Res. Public Health 11 (5), 5170–5207.

Chug, S., Kaushal, P., Kumaraguru, P., Sethi, T., 2021. Statistical learning to operationalize a domain agnostic data quality scoring. arXiv preprint arXiv:2108.08905.

Clerc, M., Daucé, E., Mattout, J., 2016. Adaptive methods in machine learning. In: Brain–Computer Interfaces 1: Foundations and Methods. Wiley Online Library, pp. 207–232.

Ditzler, G., Roveri, M., Alippi, C., Polikar, R., 2015. Learning in nonstationary environments: A survey. IEEE Comput. Intell. Mag. 10 (4), 12–25.

Evans, P., 2006. Scaling and assessment of data quality. Acta Crystallogr. D 62 (1), 72–82.

Fadlallah, H., Kilany, R., Dhayne, H., El Haddad, R., Haque, R., Taher, Y., Jaber, A., 2023. Bigqa: Declarative big data quality assessment. ACM J. Data Inf. Qual. 15 (3), 1–30.

Fan, W., Geerts, F., 2022. Foundations of Data Quality Management. Springer Nature.

Hasan, M.M., Popp, J., Oláh, J., 2020. Current landscape and influence of big data on finance. J. Big Data 7 (1), 1–17.

Hazen, B.T., Boone, C.A., Ezell, J.D., Jones-Farmer, L.A., 2014. Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications. Int. J. Prod. Econ. 154, 72–80.

Heinrich, B., Hristova, D., Klier, M., Schiller, A., Szubartowicz, M., 2018. Requirements for data quality metrics. J. Data Inf. Qual. (JDIQ) 9 (2), 1–32.

Kammerer, K., Hoppenstedt, B., Pryss, R., Stökler, S., Allgaier, J., Reichert, M., 2019. Anomaly detections for manufacturing systems based on sensor data—insights into two challenging real-world production settings. Sensors 19 (24), 5370.

Karkošková, S., 2023. Data governance model to enhance data quality in financial institutions. Inf. Syst. Manage. 40 (1), 90–110.

Karkouch, A., Mousannif, H., Al Moatassime, H., Noel, T., 2016. Data quality in internet of things: A state-of-the-art survey. J. Netw. Comput. Appl. 73, 57–81.

Kiangala, S.K., Wang, Z., 2021. An effective adaptive customization framework for small manufacturing plants using extreme gradient boosting-xgboost and random forest ensemble learning algorithms in an industry 4.0 environment. Mach. Learn. Appl. 4, 100024.

Kreuzberger, D., Kühl, N., Hirschl, S., 2023. Machine learning operations (mlops): Overview, definition, and architecture. IEEE Access.

Kumara, I., Arts, R., Di Nucci, D., Van Den Heuvel, W.J., Tamburri, D.A., 2023. Requirements and reference architecture for MLOps: Insights from industry. Authorea Prepr..

Lewis, A.E., Weiskopf, N., Abrams, Z.B., Foraker, R., Lai, A.M., Payne, P.R., Gupta, A., 2023. Electronic health record data quality assessment and tools: a systematic review. J. Am. Med. Inform. Assoc. 30 (10), 1730–1740.

Lin, J., 1991. Divergence measures based on the Shannon entropy. IEEE Trans. Inf. Theory 37 (1), 145–151.

Lionis, A., Peppas, K.P., Nistazakis, H.E., Tsigopoulos, A., 2021. RSSI probability density functions comparison using Jensen–Shannon divergence and pearson distribution. Technologies 9 (2), 26.

Liu, A., Lu, J., Song, Y., Xuan, J., Zhang, G., 2022. Concept drift detection delay index. IEEE Trans. Knowl. Data Eng. 35 (5), 4585–4597.

Liu, L., Özsu, M.T., 2009. Encyclopedia of Database Systems, vol. 6, Springer, New York, NY, USA.

Liu, S., Yamada, M., Collier, N., Sugiyama, M., 2013. Change-point detection in time-series data by relative density-ratio estimation. Neural Netw. 43, 72–83.

Mansouri, T., Sadeghi Moghadam, M.R., Monshizadeh, F., Zareravasan, A., 2023. IoT data quality issues and potential solutions: a literature review. Comput. J. 66 (3), 615–625.

McGilvray, D., 2021. Executing Data Quality Projects: Ten Steps to Quality Data and Trusted Information (TM). Academic Press.

Micevska, S., Awad, A., Sakr, S., 2021. SDDM: an interpretable statistical concept drift detection method for data streams. J. Intell. Inf. Syst. 56, 459–484.

Pipino, L.L., Lee, Y.W., Wang, R.Y., 2002. Data quality assessment. Commun. ACM 45 (4), 211–218.

Polyzotis, N., Roy, S., Whang, S.E., Zinkevich, M., 2018. Data lifecycle challenges in production machine learning: a survey. ACM SIGMOD Rec. 47 (2), 17–28.

Priestley, M., O'Donnell, F., Simperl, E., 2023. A survey of data quality requirements that matter in ML development pipelines. ACM J. Data Inf. Qual..

Reis, M.S., Saraiva, P.M., 2021. Data-centric process systems engineering: A push towards PSE 4.0. Comput. Chem. Eng. 155, 107529.

Singh, P., Suri, B., 2015. Quality assessment of data using statistical and machine learning methods. In: Computational Intelligence in Data Mining-Volume 2: Proceedings of the International Conference on CIDM, 20-21 December 2014. Springer, pp. 89–97.

Stonebraker, M., Rezig, E.K., 2019. Machine learning and big data: What is important? IEEE Data Eng. Bull. 42, 3–7.

Taleb, I., Serhani, M.A., Bouhaddioui, C., Dssouli, R., 2021. Big data quality framework: a holistic approach to continuous quality management. J. Big Data 8 (1), 1–41.

Teh, H.Y., Kempa-Liehr, A.W., Wang, K.I.-K., 2020. Sensor data quality: A systematic review. J. Big Data 7 (1), 1–49.

Wang, J., Liu, Y., Li, P., Lin, Z., Sindakis, S., Aggarwal, S., 2023. Overview of data quality: Examining the dimensions, antecedents, and impacts of data quality. J. Knowl. Econ. 1–20.

Wang, R.Y., Strong, D.M., 1996. Beyond accuracy: What data quality means to data consumers. J. Manag. Inf. Syst. 12 (4), 5–33.

Widad, E., Saida, E., Gahi, Y., 2023. Quality anomaly detection using predictive techniques: An extensive big data quality framework for reliable data analysis. IEEE Access.

Wu, H., Lin, A., Clarke, K.C., Shi, W., Cardenas-Tristan, A., Tu, Z., 2021. A comprehensive quality assessment framework for linear features from volunteered geographic information. Int. J. Geogr. Inf. Sci. 35 (9), 1826–1847.

Wu, Z., Rincon, D., Christofides, P.D., 2019. Real-time adaptive machine-learning-based predictive control of nonlinear processes. Ind. Eng. Chem. Res. 59 (6), 2275–2290.

Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S., 2016. Quality assessment for linked data: A survey. Semant. Web 7 (1), 63–93.

Zha, D., Bhat, Z.P., Lai, K.-H., Yang, F., Hu, X., 2023. Data-centric ai: Perspectives and challenges. In: Proceedings of the 2023 SIAM International Conference on Data Mining. SDM, SIAM, pp. 945–948.

Firas Bayram is a doctoral student at Karlstad University's Department of Mathematics and Computer Science. He began his Ph.D. in 2021 after obtaining an M.Sc. from the University of Genova in 2020. His research focuses on developing robust and adaptive machine learning systems, particularly in handling concept drift in non-stationary environments. Bayram also teaches courses on machine learning, software testing,

and AI engineering. His contributions to the eld are reflected in several publications addressing adaptive learning frameworks and AI-driven industrial applications.

Bestoun S. Ahmed is a professor of software engineering at Karlstad University, Sweden, and a researcher at the Czech Technical University in Prague. He received his M.Sc. from the University of Putra Malaysia in 2009, and his Ph.D. in Software Engineering from the University of Sains Malaysia in 2012. He was a postdoctoral researcher at the Swiss AI Lab (IDSIA), Switzerland, from 2015 to 2016. He has held positions as a senior lecturer and assistant professor at various institutions before joining Karlstad University in 2019, where he was promoted to full professor in 2023. Bestoun's research focuses on AI engineering, software testing, and computational intelligence.

Erik Hallin is the Head of Data Analytics & AI at Uddeholm, where he applies his extensive expertise in data science to industrial applications. With a background in machine learning and AI, Hallin is involved in various projects that leverage these technologies to optimize industrial processes. He has collaborated on multiple research publications, including work on data quality scoring frameworks and adaptive learning for machine learning applications in non-stationary environments. Hallin's contributions are pivotal in integrating AI-driven solutions within industrial settings, ensuring robust and efficient data analytics operations.