Conversational Database Management Tool: AI Driven Data Pipeline Orchestrator

Snehal Meshram
Dept. of Information Technology
YCCE
Nagpur, India
snehalmeshram1809@gmail.com

Devanshu Sanjay Markam
Dept. of Information Technology
YCCE
Nagpur, India
devanshuasm750@gmail.com

Divyanshu Chandrashekhar Tayde
Dept. of Information Technology
YCCE
Nagpur, India
divyanshutayde766@gmail.com

Falguni Shekhar Uikey
Dept. of Information Technology
YCCE
Nagpur, India
uikefalguni@gmail.com

***Abstract*—Nowadays almost every enterprise depends on both historical data and real time data which supports most of the decision making. However, such type of analysis sometimes requires expertise in SQL, distributed real time data management, and statistical structural modeling, which limits the access for users those who doesn't belong to technical field. We represents this tool as AI Driven Data Pipeline Orchestrator (AIDPO), a conversational analytics platform which developed to reduce this dependency on specialized skills. This system enables users to extract data and understand the analytical workflows through natural language by combining NL2SQL translation with real time Kafka Spark which helps to handle real time data sources, processing and time series forecasting. Also a dual assistant is integrated which separates the analytical requests from system level pipeline operations, allowing users through commands and execute both SQL and live streaming data. This tool was created using real and simulated real world datasets, achieving a forecasting RMSE of 14.23, anomaly detection precision of 0.80, and 94% accuracy of SQL selection queries . Now after analysing the final result we can say that our AIDPO can efficiently translate user inputs into understandable data operation.**

Keywords*—Natural Language to SQL (NL2SQL), Data Pipeline Orchestration, Real-Time Streaming Analytics, Forecasting, Anomaly Detection, Conversational AI.

I. Introduction

Over the past time, the natural language models and NL2SQL systems have made database querying more efficient by allowing users to converse with complex database easily [1]-[5]. After such huge progress, still most of the existing NL2SQL systems focuses on static datasets and provides limited support for real time data stream processing or multi-stage analytical workflows. At the same time, data engineering platforms such as Apache Kafka and PySpark offers scalable streaming analytics and large scale processing [8], [9]. While practical deployment, constructing and managing these technology which requires specialized knowledge, including manual pipeline orchestration and operational tuning.

To overcome these limitations, this system introduces the AI Driven Data Pipeline Orchestrator (AIDPO), a conversational tool developed from several inspection we found that most of the analytical tasks begin with a simple question rather than technical workflow. Instead of influence users no technical background to adapt such database schemas, query languages, or pipeline configurations, AIDPO adapts the system behavior in such way that users naturally describes their purpose. Mostly users asked in plain english and resolved through integrated database access, real time data, within the backend infrastructure.

This system comes with a dual assistant that distinguishes the analytical interactions from system level control. One assistant handles data exploration and schema related requests, while the second is responsible for helping in pipeline execution which is real time or live data, and processing tasks. These two assistants allows users to communicate properly and differentiate between real time and static data, while ensuring that execution of each request must predict properly. Together, these assistants enable AIDPO to support practical analytics purpose that uses historical datasets as well as continuously generated real time data.

II. Ease of Use

Many users still struggle to work with modern technology and data systems because recent analysis assumes that familiarity with the basic query languages and data engineering tools. While development of this AIDPO tool, usability was treated as a primary feature rather than an future feature. Users can easily interact with the tool by telling tool what they want to extract, such as data accessing, query formulation, and real time data execution etc., are handled internally and effciently. This technique reduces the direct experience to technical complexity and allows non tech users to obtain and grab analytical results without engaging with SQL syntax or complex system languages.[20].

Although, it became clear that many tasks fails but not because of data quality, but due to the need of connection between different tools. Our tool also allows users to switch between query interfaces, streaming frameworks, and visualization platforms, which increase the user experience and reduce the misunderstanding between tool and users. Our AIDPO is mainly handled internally, so users doesn't required to start, stop, or manage individual components. As a result, repeated queries can be performed with less steps, and similar queries can also produce stable results across different use cases according to user.

In AIDPO, interaction between user and system and also the backend execution is handled individually to avoid mixing of several requests . Their are some tasks such as querying data and interpreting results are redirected to the Data Assistant, whereas pipeline execution, Kafka stream handling, and PySpark job control are managed by the another assistant.

Users are not allowed to raise commands or manage execution steps, instead they interact with the system through short messages, which helps keep interactions simple while preserving reliable control over analytical operations [20].

III. System Overview

On the AIDPO platform, user requests go through more than one internal path. The requests may be for data access or explaining results are handled in a different way, the actions that control execution or manage streams work in different way. These differences inside our system doesn't show up in the user interface and user doesn't need to know about them. From the user's point of view, everything looks same, but the actual work is split into different parts of the system depending on the type of task and user requirements.

A. System Goals

AIDPO aims to delete the technical barrier between non technical users and complex data enginnering workflows. Our goal is to help non tech users to retrieve, process, analyze data using plain english language rather than SQL generation (NL2SQL) [1]-[5], pipeline orchestration, and a perfect interface for multiple streaming operations.

AIDPO supports multiple core and useful functions, including conversational interaction with structured PostgreSQL. Plain english language inputs are then processed and transformed into executable SQL queries [1]-[3]. And then it supports the efficient management of live data pipelines built on Apache Kafka and PySpark, allowing real time ingestion and transformation, and perfect insight generation without manual configuration [8], [9]. Third, our system also comes with integrated forecasting and anomaly detection modules that process the time series data to identify trends, future values, and unsual patterns [11]-[17]. After all a final analytical results are delivered using dashboards and tabular format, which makes interpretation much easier for users without a technical background [18], [19].

B. Overall Workflows

The operational workflow of AIDPO begins when the user creates their profile and set their profession and start storing its data , after sufficient amount of data is stored in database then user can easily communicate with our system, the user can submits a natural language query or command through the conversational interface. Then the system's engine determines whether the request corresponds to a database query or not, or a streaming pipeline action, a forecasting task, or an anomaly detection operation [20]. For the database related tasks, the NL2SQL module performs schema linking, query construction, and validation before executing the translated SQL query on PostgreSQL [1]-[3].

Most of the streaming workflows depends on predefined Kafka producers and PySpark streaming jobs which are assisted by the DevOps Assistant. These components are able to handle live data streaming and processing before execution efficiently [8], [9]. Then the processed data further delivered to the analytical layer, where forecasting and anomaly detection may takes place depending on the user's intent [11], [17]. Then the results whether numerical outputs, alerts, or trend analysis they are converted into automated dashboards, graphs, and tabular summaries [18], [19].

Now AIDPO can manage user interaction and those two separate assistant very efficiently and easily, which increase the performance and user experience.

IV. System Architecture

AIDPO architecture is divided into multiple components which handle natural language analysis, data, live streaming execution, and result evaluation. And also, batch processing and real time data streaming is handled using different conversational components, even though these depends on different backend components. Basically different parts of the systems handle their tasks separately rather than depending on shared execution logic which is used commonly. Changes made to the one part of the implementation doesn't require adjustments in other areas also, as interactions are limited to predefined points of communication. Whereas during testing, failures are observed to remain confined to the component where they originated, allowing other parts of the pipeline to continue functioning without failure. (see **Fig. 1.**)

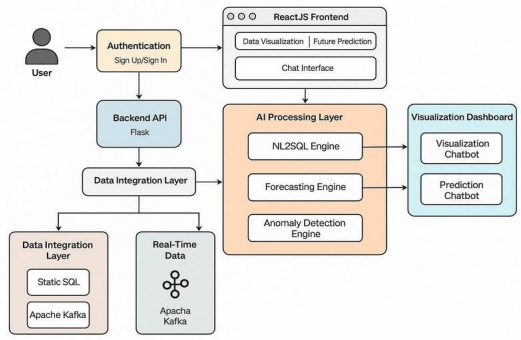

Fig. 1. Overall system architecture of AIDPO

A. Conversational Interface Layer

Mainly users interact with AIDPO through a conversational interface that separates analytical requests from execution related operations. Queries related to data access and result interpretation are handled by the Data Assistant, while tasks such as pipeline control, kafka producer management, and PySpark job execution are handled by the DevOps Assistant [20]. As a result, mostly users interact only with analytical requests, rather than live execution .

User inputs are first passed through an specific classification module that control the nature of the request, forecasting anomaly detection, or live data streaming pipeline control. Changes in operation type is managed internally by this layer, because this layer does allows exposing system internals or external command structures to the user [20].

B. NL2SQL and Query Processing Layer

Once any user submits an query, first the NL2SQL model performs tokenization then entity recognition and then schema linking, to map natural language terms to actual database attributes. Our system comes with a context-aware parser which help to generate SQL queries, which are then further validated to ensure syntactic and semantic errors [1]-[5].

Especially this layer's work is to handle interaction between database internally, and ensures that users can only query about relational datasets through natural language. It also include safeguards against unsafe queries, for which maintaining database integrity and reducing the risk of accidental system misue is very important [1]-[3].

C. Streaming and Pipeline Orchestration Layer

For real-time data streaming, Apache kafka is used as the main backbone and PySpark is used in structured streaming

services as a processing engine. When users requests about streaming operations through the conversational interface, the Assistant provides producer service, sets up kafka topics, and launches PySpark jobs [8], [9].

Kafka is used for high volume event handlin and PySpark performs separate batch processing and applies streaming aggregations, anomaly detection, or forecasting models as needed [8], [9].

Then at last the processed data retained for later analysis and made a direct decision as real time insights which requires manual pipeline configuration. This orchestration layer manages condition of pipeline and processing latency to help and maintain the stablility of execution under complex workload conditions.

D. Visualization Layer

AIDPO includes various analytical modules which is used for forecasting, anomaly detection, and trend interpretation as a part of data analysis. In the current implementation, forecasting is carried out using statistical or machine learning models, including ARIMA and LSTM, to produce a short-term prediction [11], [14]. Anomaly detection depends on multiple statistical performance measures, such as Z-scores, to identify outliers and drastic changes in observed results [15], [17].

This visualization module automatically selects the appropriate graph types or char types such as line charts, bar charts, pie charts or trend curves based on the structure and schema of the user's query. Then the results are displayed on dashboard with summarized results, highlighted anomalies, and future predictions. This layer also ensures that users must communicate clearly and efficiently [18], [19], enabling users to make proper decisions.

V. METHODOLOGY

The methodology which endorse in AIDPO mainly focuses on mapping user intent against executable analytical workflows across both historical and streaming data components. The system also combines natural language processing and database querying and live data streaming, forecasting, and visualization of unified operational pipeline. This section presents the methodological steps which helps in handling the user queries and production of usefull insights.

A. Natural Language Understanding

When any user asks a question, the system will starts processing the input using various natural language processes. The text is broken into various parts called tokens and then find the important words, names and patterns. A classifier figures out , which kind or what kind of request from user, find unusual patterns , or running a process. This helps to sent the request to the right part of the system and avoids confusion in understanding [20]. This module also checks if the user's query matches the system which handle the data , making sure that everything works safely and effectively [20]. This module also verifies that user's request must aligns with available database schemas and capabilities and ensuring the strength and operational safety.

B. NL2SQL Translation and Database Execution

For multiple analytical queries including relational data, and NL2SQL module involves in translating user queries into SQL statements. Schema linking for natural language terms with specific database columns and tables, while context based parsing generates query structures [1]-[5]. This system analyse these components and selects the most suitable query, and applies safety checks to maintain minimum wrong operations [1]-[3]. After multiple stages of validation, the final query is further executed on a PostgreSQL terminal, and then gives output into a structred table for better understanding and visualization. All these stages allows users to work with relational database and live data without manually writing any SQL queries.

Additionally, NL2SQL also applies query optimization techniques which reorganizes the generated statements to improve the efficiency of execution on the large datasets. This system uses queries which is only valid SQL expressions when feasible. The enhanced data processing pipeline also supports the consistent query performance while preserving a natural and intuitive interaction experience for the users. For example, if any user asks, "Show the average salary of employees in the IT department," then our system identifies relevent attributes and generates SQL query respectively.

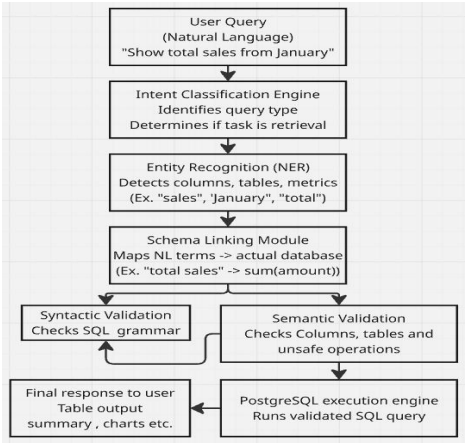

Fig. 2. NL2SQL Workflow showing intent parsing, schema mapping.

C. Streaming Pipeline Activation and Real Time Processing

Now for real time data workflow, the system always follows an automated orchestration method. Detecting a streaming related content, the Assistant initiates Kafka producers and configures topics, and starts PySpark Structured Streaming jobs [8], [9]. Kafka is here to handle the incoming events and PySpark is allowed to apply proper operations such as filtering, computation or user defined analytical operations. Although, this model helps in low latency and allowing users to understand the real-time trends . Pipelines states that the processing metric is monitored continuously to ensure stable and high throughput execution [8], [9]. The real time orchestration is shown in **Fig. 3.**

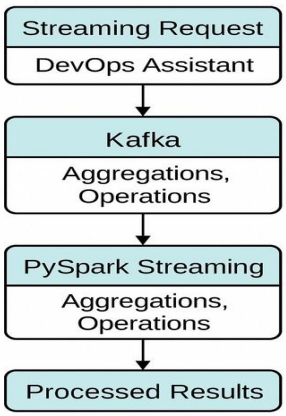

Fig. 3. Streaming pipeline using Kafka and PySpark

D. *Anomaly Detection, and Visualization*

When a forecasting or anomaly detection is requested by the user, AIDPO applies it's mathematical models like ARIMA for time series, or Z-score based evaluation for identifying abnormal patterns. All these integrated models are trained using historical data which is retrieved from the database or streaming outputs. Also forecasting the curves or trend lines are generated and then passed to the main layer which is visualization layer, which selects particular chart types based on the characteristics of the data [18], [19]. The final result displays various graphs and charts and textual content by which users can easily visualize the model behavior and system nature. This whole design and workflow ensures that the complex analytical results are delivered in an accessible and user friendly format.

$$y_t = c + \Sigma(\varphi_i \cdot y_{t-i}) + \Sigma(\theta_j \cdot \varepsilon_{t-j}) + \varepsilon_t \quad (1)$$

$$Z = (x - \mu)/\sigma \quad (2)$$

VI. Results And Evaluation

After our evaluation of AIDPO we found that, we need to focus on these three core components: first is accuracy of natural language to SQL translation and the responsiveness of the real time streaming pipeline, and the effectiveness of the forecasting and anomaly detection modules. The system is tested on multiple types of queries and live kafka-PySpark real time data, and historical data to check the performance and robustness, and practical usability. Overall this section helps to display the experimental outcomes and highlight the system's operational growth.

Second the NL2SQL component which indicate the high reliability across all common analytical query types. The main performance was measured was compared automatically using generated SQL statements with manually validated actual queries. The accuracy is still consistently high for selection, filtering, and aggregation queries, while other complex nested and join queries produced slightly lower accuracy due to structural puzzle in natural language [1]-[5]. Table I. Summarizes the overall results of the NL2SQL evaluation conducted on the test dataset.

TABLE I. NL2SQL EVALUATION RESULTS

QUERY TYPES	EXAMPLES	ACCURACY
Selection Queries	Select column FROM table	94 %
Filtering Queries	WHERE conditions	87 %
Aggregation Queries	SUM, AVG, COUNT	90 %
Join Queries	Multi table joins	82 %
Complex Nested Queries	Sub-queries, multiple clauses	75 %

Fig. 4. Visual representation using a bar graph for sample analytics output

As we inspect the live data streaming performance and observe Kafka-PySpark pipelines which was evaluated under various conditions and also by increasing the flow rate. The system also perfectly maintained the low processing latency across all over the tested throughput stages, showing its ability to observe real time analytics properly. Fig. 5. shows the relationship between flow rate and end-to-end processing latency which also displays the stable performance even after higher data volumes. These results shows that the orchestrated pipeline manages effectively batch processing without compromising the timelines.

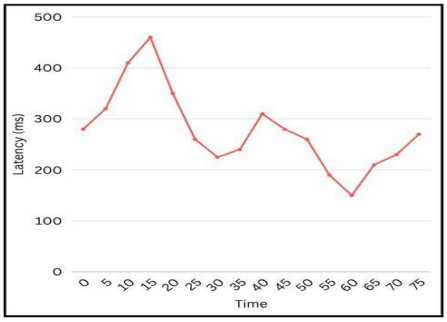

Fig. 5. Streaming Pipeline Latency

AIDPO forecasting module is tested using historical time series data. The prediction using ARIMA , the engine produced stable and proper forecasts results with minimal deviation from actual observed values [11]-[14]. This trend lines and projected values were visualized for user evaluation, as show in Fig. 6. This detection of anomaly, which uses Z-score deviation to spot deviation [15]-[17], successfully flagged uncommon values across multiple datasets. Fig. 7. shows the ability of system to recognize both anomaly and extended segments. Jointly, the forecasting and detection of anomaly results confirm that the AIDPO can gain actionable awareness from both historical and live streaming data.

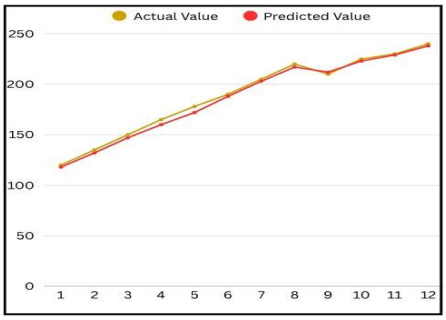

Fig. 6. Forecasting output showing actual versus predicted values

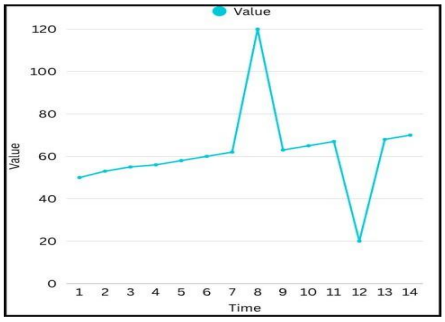

Fig. 7. Anomaly detection results showing detected spikes and drops

Overall, the experimental findings finally demonstrate that AIDPO offers a reliable and efficient conversational analytics. The combination of query translation, real time data pipeline performance, and effective analytical models validates the ability of system for organizations seeking to democratize to access the data driven insights while minimizing technical barriers.

VII. Comparison With Existing Work

The existing research in natural language querying and automated data analytics has produced multiple tools that clarify the database interaction, streaming pipeline executions, or enable conversational interfaces. However, these systems usually address the isolated components of the data lifecycle and do not provide an integrated or end to end orchestration framework [1]-[5], [8], [9], [18]-[20]. Our AIDPO pick out itself by simply unifying NL2SQL translation, real time streaming operations, forecasting, anomaly detection, and automated visualization within a single conversational environment.

As the NL2SQL systems primarily focus on converting natural language input into syntactically correct SQL statements [1]-[5]. Which is very effective for data extraction which is in structured format, they do not include the mechanisms for activating streaming pipelines or applying any advanced analytical tasks such as forecasting or anomaly detection. But in AIDPO it bridges this gap by linking classification with both databases and pipeline oriented execution flows, enabling users to communicate seamlessly between historical query analysis and real time data processing [20].

Similarly, existing streaming platforms based on Apache Kafka and PySpark offer strong support for the high throughput event processing, but they require specialized configuration and domain knowledge [8], [9]. These platforms are commonly controlled through multiple command line interfaces or scripts, which brake the accessibility for non technical users. The AIDPO overcomes this limitation through the various Assistant, which abstracts operational tasks and enables the streaming workflows to be executed through the natural language, significantly reducing the cognitive load on users [20].

This conversational business intelligence system has also emerged, offering natural language dashboards and automated evaluation [18], [19]. However, such systems mainly rely on the predefined datasets and lack the adaptability which needed to manage dynamic streaming workflows. Our AIDPO widen beyond this , we have dashboard driven tools by integrating real time processing capabilities, flexible pipeline activation, and predictive modeling through ARIMA based forecasting and statistical anomaly detection [11], [17]. This combination allows to respond to the diverse analytical demands all across both static and evolving datasets.

Overall, our AIDPO provides a more complete and major solution than existing systems by integrating natural language interaction with interactive data engineering and analytics options. Our platform removes the gap between accessibility and automation, making it more suitable for non tech users and various organisations using conversational interface for complex analytical operations.

VIII. Future Work

For all that our AIDPO illustrate a strong performance across all natural language interpretation, streaming analytics, and automated visualization outputs, several components can further developed its capabilities. One desired direction is the alliance of more advanced language models capable of handling highly complex and strong analytical queries, multi intent statements, and conversational memory over extended data. Improving the robustness of schema and understanding across diversified databases would also increase the adaptability in enterprise level environment where structures changes frequently.

The real time analysis can be developed by integrating more complex forecasting techniques and anomaly detection algorithms, particularly those leveraging deep learning architectures, like expanding support for additional streaming platforms beyond Kafka and PySpark such as Flink which would enhance the applicability of system in multiple operational settings. For future idea we may cover data quality checks componets or automated feature retrieval system for time data analysis, and dynamic streaming pipeline enhancement to reduce latency when high load is integrated.

For the experience of users, we would incorporate the multimodal interaction such as voice commands or graphical query builders could further simplify analytics for non technical user. Enhances the dashboard customization and user experience, collaborative features, and automated report generation would also provide a greater flexibility for various decision makers. In addition, establishing audit trails or role based access control, and encryption mechanisms would improve the security and encryption for sensitive datasets.

Overall, these extensions can transform AIDPO into a more intelligent, scalable, and secure analytics assistant which is capable of supporting complex data environment in real world deployments.

IX. Conclusion

This AI-Driven Data Pipeline Orchestrator (AIDPO) introduces a best conversational tool that just simplifies the interaction with complex data systems and automates end to end analytical dataflows. By just integrating a natural language input interpretation and SQL generation, real-time streaming analytics, forecasting models, and automated visualization, this system effectively bridges the gap between technical data sources and non technical users. The dual assistant interface increases the accessibility by just showing users through both the query formulation system and insight review system, enabling inherent analytics without any requiring knowledge of SQL, data pipelines, or statistical modeling.

Experimental evaluation shows that our AIDPO execute very reliably across over multiple dimensions, including the accuracy of query translation, live data streaming throughput, quality of forecasting, and detection of anomaly. The

orchestration of Kafka and PySpark provides a scalable foundation or base for handling various dynamic and high velocity data, while the visualization engine delivers clear and actionable patterns. These final results bring out the AIDPO's ability as an intelligent assistant for data analysis, monitoring, and decision support in environment that depends on continuous and live data flow.

Overall, this system presents a significant step towards democratizing data analytics by just simple combining the conversational AI with an automated data engineering processes. With future improvement in model development, multimodal inter activity, security, and flexibility, AIDPO can evolve into a comprehensive platform for real time interactive, and intelligent data analysis within modern organizations.

ACKNOWLEDGMENT

The authors express their sincere gratitude to the Department of Information Technology, Yeshwantrao Chavan College of Engineering, Nagpur, for providing the resources and academic support which is necessary to complete this project. This team also acknowledges the guidance of faculty members and all other supporting staff , whose insights helped us a lot to shape the technical direction of this project. Their encouragement and feedback and support played an important role in the success and development of the AI-Driven Data Pipeline Orchestrator.

REFERENCES

[1] X. Zhang et al., "A comprehensive survey on natural language to SQL translation," ACM Comput. Survey, 2023.

[2] K. Xu et al., "RAT-SQL: Relation-aware schema encoding for text-to-SQL parsers," Proc. ACL, 2020.

[3] V. Zhong, C. Xiong, and R. Socher, "Seq2SQL: Generating structured queries from natural language using reinforcement learning," arXiv, 2017.

[4] B. Bogin, M. Gardner, and J. Berant, "Representing schemas with graph neural network for text-to-SQL parsing," ACL, 2019.

[5] H. Yu et al., "SyntaxSQLNet: Syntax tree network for complex and cross domain text to SQL generation," EMNLP, 2018.

[6] A. Thusoo et al., "Hive: A warehousing solution over a map-reduce framework," Proc. VLDB, 2010.

[7] T. Akidau et al., "The dataflow model: A practical approach to balancing correctness, latency, and cost," Proc. VLDB, 2015.

[8] M. Armbrust et al., "Structured streaming: A declarative API for real time application in Apache Spark," Proc. SIGMOD, 2018.

[9] J. Kreps et al., "Kafka: A distributed messaging system for log processing," NetDB, 2011.

[10] Apache Software Foundation, "Apache Kafka Documentation," 2024.

[11] S. Makridakis et al., "Statistical and machine learning forecasting methods: Concerns and ways forward," PLOS One, 2018.

[12] R. Hyndman and G. Athanasopoulos, Forecasting: Principles and Practice, Otexts, 2020.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, 1997.

[14] C. Chatfield, The Analysis of Time Series, 6th ed., Chapman and Hall, 2004.

[15] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Comput. Surveys, 2009.

[16] S. Gupta et al., "Outlier detection for temporal data: A review," IEEE TKDE, 2014.

[17] N. Laptev et al., "Time-series anomaly detection using deep learning," Proc. ACM KDD Workshop, 2015.

[18] A. Halevy, P. Norving, and F. Pereira, "The unreasonable effectiveness of data," IEEE Intell. Syst., 2009.

[19] P. Deutsch, "Data engineering for intelligent systems," IEEE Software, 2022.

[20] S. Amershi et al., "Guidelines for human-AI interaction," CHI, 2019.