INFORMATION SYSTEMS AND TECHNOLOGIES

DOI https://doi.org/10.30525/978-9934-26-388-0-1

RNN IMPLEMENTATION FOR STREAMING DATA ANALYSIS AND PREDICTION

ЗАСТОСУВАННЯ РЕКУРЕНТНИХ НЕЙРОННИХ МЕРЕЖ ДЛЯ АНАЛІЗУ ТА ПРОГНОЗУВАННЯ ПОТОКОВИХ ДАНИХ

Ilin M. O.
Postgraduate Student at the Department of Computer Systems Software, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" Kyiv, Ukraine

Ільїн М. О.
аспірант кафедри програмного забезпечення комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» м. Київ, Україна

Oleshchenko L. M.
Candidate of Technical Sciences, Associate Professor at the Computer Systems Software Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" Kyiv, Ukraine

Олещенко Л. М.
кандидат технічних наук, доцент кафедри програмного забезпечення комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» м. Київ, Україна

In our rapidly evolving digital landscape, the generation of streaming data has become ubiquitous. From social media updates and financial market fluctuations to sensor readings in IoT devices and real-time monitoring of industrial processes, the volume of data generated and transmitted in real-time is staggering (fig. 1).

Change Data Capture (CDC) streaming tool is using for aggregating data sources, connecting to relational databases or transactional systems, whether on-premises or in the cloud. These sources are then linked to a stream processor.

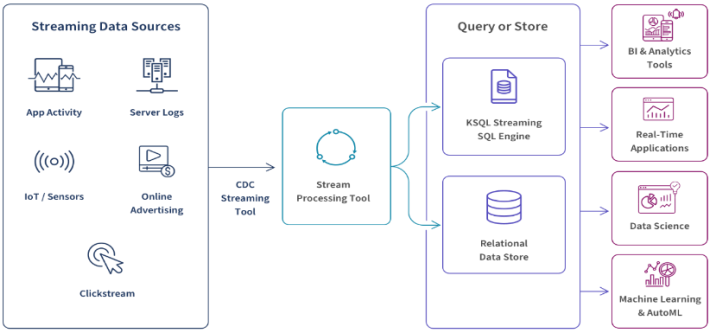

Fig. 1. Traditional streaming data processing tools [1]

It is important to create a stream processor with scalable, fast, fault-tolerant, and integrated tools like Apache Kafka or Amazon Kinesis. Data is processed sequentially and incrementally, often on a record-by-record basis, or through sliding time windows.

Tools like Google BigQuery, Snowflake, Amazon Kinesis Data Analytics, or Dataflow are using to perform various analytics (e.g., filtering, aggregation) on the streaming data. Two approaches are available:

1. Query the data stream in real-time using KSQL or ksqlDB, a streaming SQL engine.
2. Store the data for later querying, with storage options like Amazon S3, Amazon Redshift, and Google Storage.

Streaming data output serves multiple purposes, including BI and analytics, data science, and machine learning. BI tools enable interactive data visualization, real-time dashboards, and alerts, aiding in detecting anomalies and fraud. Data scientists can apply algorithms in-stream, avoiding the need to wait for data to be stored, enabling real-time analysis. Machine learning and AutoML models can leverage incremental learning libraries like Creme for sequential data processing and predictions.

In data-rich environment, the ability to analyze and predict trends, anomalies, or events becomes increasingly vital for businesses, researchers, and decision-makers. One powerful tool in this realm is the recurrent neural network (RNN).

RNNs are a class of artificial neural networks specially designed to handle sequential data. Unlike traditional feedforward neural networks, RNNs have a unique architecture that allows them to capture dependencies within data that unfolds over time. This makes RNNs exceptionally well-suited for analyzing and predicting patterns in streaming data, as they can take into account the temporal context of the data [2–4].

A variation of RNNs, known as Long Short-Term Memory (LSTM), has gained immense popularity for its ability to capture long-range dependencies in data. LSTMs incorporate memory cells that can store and retrieve information over extended time intervals, making them highly effective for tasks like natural language processing, speech recognition, and time-series forecasting.

The continuous operation of the LSTM module in tandem with Kafka Streams facilitates the ongoing updating of its model as new data becomes available. This dynamic updating capability enables the predictions to adjust to changing conditions over time. It's crucial to acknowledge that although LSTMs offer numerous benefits for handling and predicting streaming data, they do come with certain challenges. The computational demands can be high, and real-time training with extensive datasets may necessitate specialized hardware or distributed computing resources. Furthermore, the selection of the appropriate architecture and hyperparameters is paramount in ensuring optimal performance for a given streaming data application (fig. 2).

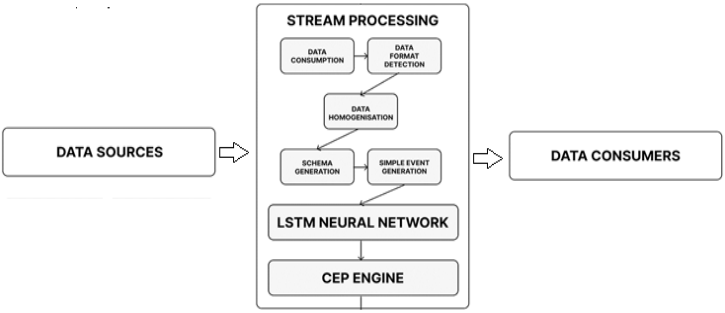

Fig. 2. Proposed streaming data analysis architecture using LSTM

CEP (Complex Event Processing) technology allows real-time processing and analysis of high volumes of streaming data to identify complex patterns, trends, and relationships within that data. A CEP engine is the core component of a CEP system, responsible for efficiently and effectively processing continuous streams of events and identifying meaningful patterns or conditions. A CEP engine focuses on processing events, which are occurrences or updates in a system that have relevance for analysis. These events could be generated from various sources such as sensors, logs, transactions, or other data streams. CEP engines operate based on predefined rules or queries. These rules define the conditions or patterns that the engine should identify within the streaming data. The rules are often expressed using

a specialized language designed for event processing, such as Event Processing Language (EPL) or SQL-like queries. CEP engines handle both temporal and spatial correlation of events. Temporal correlation involves understanding the timing relationships between events, while spatial correlation involves understanding the spatial relationships or patterns within the data. The engine can recognize complex patterns in the streaming data, including sequences of events, combinations of events, and correlations between different types of events. This ability to identify patterns in real-time is crucial for detecting complex situations or conditions as they unfold. CEP engines often integrate with external systems, databases, and applications to enrich event processing. This integration allows for more comprehensive analysis by incorporating additional context or historical data. CEP engines find applications in various domains, including finance for fraud detection, telecommunications for network monitoring, supply chain for tracking and optimization, and IoT for monitoring and controlling devices in real-time.

RNNs require preprocessing to adapt streaming data into a suitable format. Data is usually organized into overlapping windows or fixed-sized chunks to ensure a continuous flow of information. The success of RNNs in streaming data analysis depends significantly on feature engineering. Relevant features need to be selected and extracted from the streaming data to facilitate accurate predictions. Streaming data often requires real-time or near-real-time analysis. RNNs can be deployed in streaming data pipelines to provide instant insights and predictions. This can be especially valuable in applications such as fraud detection, network monitoring, and predictive maintenance. RNNs excel in identifying anomalies or unusual patterns within streaming data. They can learn normal behavior over time and flag deviations from the norm. This is crucial for detecting security breaches, equipment failures, or any unexpected events in various domains. RNNs can be designed for continuous learning, adapting to changing data patterns as they evolve over time. This adaptability makes them a valuable tool for staying up-to-date with dynamic streaming data.

In conclusion, the analysis and prediction of streaming data have become indispensable in many fields, and RNN have proven to be a robust solution for this challenging task. Their ability to capture temporal dependencies and adapt to evolving data makes them a valuable tool in scenarios where traditional machine learning models fall short. As streaming data continues to proliferate in our data-driven world, the adoption of RNNs is likely to play an increasingly significant role in enhancing decision-making, improving efficiency, and driving innovation.

Bibliography:

1. Streaming Data. URL: https://www.qlik.com/us/streaming-data.

2. Duda P., Jaworski M., Cader A., & Wang L. On training deep neural networks using a streaming approach. Journal of Artificial Intelligence and Soft Computing Research, 2019. 10(1), pp. 15-26. DOI: 10.2478/ jaiscr-2020-0002.

3. Bilal Jan, Haleem Farman, Murad Khan, Muhammad Imran, Ihtesham Ul Islam, Awais Ahmad, Shaukat Ali, Gwanggil Jeon. Deep learning in big data analytics: a comparative study. Computers & Electrical Engineering, 2019, Volume 75, pp. 275-287. https://doi.org/10.1016/j. compeleceng.2017.12.009.

4. Tariverdi A., Venkiteswaran V.K., Richter M., Elle O.J., Tørresen J., Mathiassen K., Misra S. and Martinsen Ø.G. A Recurrent Neural-Network-Based Real-Time Dynamic Model for Soft Continuum Manipulators. Front. Robot. AI, 2021, 8, Article 631303. DOI: 10.3389/frobt.2021.631303