Chapter 2
Data Engineering for the Factory of the Future, Multimedia Applications and Cyber-Physical Systems: Part 2 - Algorithms and Python-Based Software Development for Time-Series Data Format Conversion

Emmanuel Oyekanlu
Corning Incorporated, New York, USA

David Kuhn
Corning Incorporated, New York, USA

Grethel Mulroy
Corning Incorporated, New York, USA

ABSTRACT

This chapter is the companion chapter to "Part 1: State-of-the-Art Time-Series Data Formats Performance Evaluation." In this chapter, algorithms for converting data from one format to other formats are presented. To implement the algorithms, existing open-source Python libraries are used extensively, and where needed, new Python routines for converting data formats are developed. It is envisaged that the algorithms and Python libraries and routines that are freely provided in this chapter will be useful for data engineers, data scientists, and for industrial IoT, cyber-physical systems (CPS), multimedia, and big data practitioners who are on the quest to use different types of data formats that are compatible with memory-constrained factory floor IoT devices. It will also be useful for Delta Lake and big data engineers, who are on the quest for delivering robust bronze, silver, and gold data lakes in the cloud.

DOI: 10.4018/978-1-7998-7852-0.ch002

INTRODUCTION

In Part I (Oyekanlu et al., 2022), an extensive study regarding the benefits of using different types of existing legacy and state-of-the-art data formats was conducted. In this chapter, algorithms that can facilitate data formats conversion, metadata generation, storage footprint reduction, schema design, schema integration and schema evolution for time series data are provided. Based on extensive literature search, this chapter provides the most extensive but detailed algorithms regarding how different types of data formats can be changed from one format to another. Python implementations for the algorithms are also provided. Our contribution in this chapter will enable Data Engineers, Software Engineers, Cyber-Physical Systems (CPS) Engineers, industrial Multimedia Content Developers, Data Scientists, Cloud Engineers, DevOp Engineers, IoT and Big Data practitioners, etc., to easily and affordable be able to enjoy the benefits of using the different types of available data formats for time series data analytics and for time series-based AI applications.

Without loss of generality, the usefulness of our algorithms, software implementation, and data format conversion approaches in this chapter can be extended to other types of data that are different from time-series data sets. It is instructive to mention that, due to our usage of Python, different type of available IoT devices may be able to easily use our algorithms and software implementation codes to work with different type of data formats for smart manufacturing, medical, multimedia and CPS applications.

Researchers have always been on the quest for providing easy and affordable means by which different data sets can be made more interoperable. Methods of robustly converting data from one form to another is also in high demand. In (Pivarski et al., 2020), authors present Awkward Array, a Python-based, Numpy-like interface that can be used to handle JSON-like data in similar ways by which Numpy handles rectilinear arrays of numbers. Awkward Array is a generalization of Numpy's core function to cater to the needs of nested record, variable-length lists and most other data sets that have JSON-like constructs. Awkward Arrays can also be effectively used to handle JSON-like data with columnar structures. Authors in (Belov et al., 2021) presented a comparative analysis of Avro, comma separated values (CSV), JavaScript Object Notation (JSON), Optimized Row Columnar (ORC) and Parquet data formats on Apache Spark framework. Results of an evaluation of these data formats in terms of volume, and processing speeds for different analytics operations such as sorting, grouping, reading unique values, and filtering operations are also presented.

In (Ahmed et al., 2017), researchers explore different data formats that are appropriate for use on Big Data platforms. Results of the analysis presented in (Ahmed et al., 2017) can be used by different practitioners to select data formats that are most suitable for their project needs. In (Ye et al., 2022), authors discuss the Asset Administration Shell (AAS), a new approach for implementing data interoperability across different CPS automation pyramids. The AASX data format, which represents AAS information; and supports data communication via CPS Open Platform Communications United Architecture (OPC UA) was also presented. Also, in (Ye et al., 2022), an AASX-based solution for bidirectional data exchange between enterprise and control applications in CPS was discussed. An approach for interoperable data format exchange between AASX and Excel spreadsheets applications was also presented.

In (Oyekanlu et al., 2017), authors developed CSV2RDF, a protocol that focuses on how Semantic Web technologies are used to convert CSV data into Resource Definition Language (RDF). Techniques by which CSV data can be parsed into RDF triples are also discussed. As discussed by researchers in (Oyekanlu, 2017; Oyekanlu, 2018a; Oyekanlu, 2018b; Oyekanlu, 2018c; Oyekanlu, 2018d; Oyekanlu, Onidare, & Oladele, 2018; Oyekanlu & Scoles, 2018a; Oyekanlu & Scoles, 2018b; Oyekanlu, Scoles, &