Abstract

Phasor Measurement Units (PMUs) provide us high resolution time-series data about the status of the electric grid, but occasionally PMU malfunctions or external factors lead to data being reported that isn’t reflective of the actual state of the grid. In this project, we propose using DISTIL in conjunction with a Python API to detect potential data quality issues and make that information accessible to users.

Introduction

The electric grid is a massive system of many interconnected parts that all need to work simultaneously in order to bring us electricity. Usually, these parts work together very well, but when something goes wrong, the consequences can be disastrous, as can be seen in the prevalence of wildfires caused by downed utility lines in just the past few years. One way to avoid such consequences is to have real-time information about what is happening in the power grid, so any faults can be detected and corrected hopefully before they pose significant concern. PingThings’ PredictiveGrid™ platform provides a way to visualize and analyze such data through both historical and real-time information about the status of the electric grid, but in practice the resulting analytics are only as good as the data streaming in from the various phasor measurement units (PMUs), which report phase and angle data of the voltage and current on the grid. When PMUs stop reporting data accurately (if they go offline, for example), they will often spit out erroneous values that can significantly throw off any analytic calculations. For example, a sensor sometimes outputs the same value repeatedly when it gets taken offline, which can make it appear like the phase angle difference between two locations in a network is changing rapidly when in fact it is staying roughly the same. In order to avoid such incorrect conclusions, it is necessary to have a way of differentiating genuine data points from erroneous ones. Furthermore, this information about the quality of the data must be easily accessible so that users of the platform can efficiently apply the information to their specific use case, and must be generated fast enough to keep up with real time data as well as historical data.

Materials and Methods

We picked our criteria for detecting potentially erroneous data based on the findings of previous research from the data science team at PingThings, which determined some common signs of incorrect data. We decided to implement these data quality checks with DISTIL®, a system for efficiently processing synchrophasor data in the PredictiveGrid™. DISTIL is especially useful in that it can both analyze a lot of historical data relatively quickly, as well as keep up with real time changes to the data. Our approach uses DISTIL to generate separate streams containing data quality information for each stream where data quality analysis is desired, specifically detecting repeated values, duplicate timestamps, and outputs of exactly zero in the input data. Each of these output streams are updated as new data streams in, and only contains data points when an issue has been detected, to minimize the memory storage of these auxiliary streams.

In order to make this data easily accessible to the users, we developed a Python API specifically for this data quality information, which would allow users interested in the data quality of a particular stream to easily view the kinds of data quality checks that have been run on that stream. Each output stream contains an attribute linking it back to the original stream, which the API uses to group streams together into a single set. The user can operate on the original stream as normal, but can also view which types of data quality information are available, and query a specific time window to check whether there were any potential data quality issues flagged in that interval.

Results

Overall, this approach was successful at generating streams that flagged data quality issues in both historical and streaming data (see Fig. 1), and making it simple to interact with that information through Python. However, while we were able to implement the other data quality checks, we were unable to implement gap detection. This is because DISTIL operates by detecting where the input stream has changed, while a gap represents a portion of the input stream that has never changed from its initial, empty state. Implementing this within the framework of our approach would require making somewhat significant changes to DISTIL, which was outside the scope of this project.

Next Steps

Based on these results, one next step would be to investigate how to implement gap detection. Additionally, although we have some data quality checks that are functioning, this is by no means an exhaustive set of criteria, so further research is needed to create a more robust set of data quality checks. Additionally, while there exist data quality logs for some streams using the C37.118 protocol, the information contained within them is somewhat difficult to parse, as there are a range of different types of issues that are all combined into a single log. It could be useful to separate the information contained within these streams into a set of different data quality streams, and incorporate that information into a more usable format.

References