Data Cleaning in the Era of Data Science: Challenges and Opportunities

El Kindi Rezig
elkindi@csail.mit.edu
MIT CSAIL

There is a decades-long literature on data cleaning [1]. However, existing systems are wired to address scenarios as follows: We have a dirty dataset D and a tool T to clean it. We run T on D and get a clean instance of D. While this scenario was acceptable when data pipelines were mostly fixed (e.g., ETL), it falls short when it comes to data science applications. Nowadays, data scientists don't have fixed pipelines and they constantly experiment with new ones. This means that manually tuning one pipeline and expecting it to work for all applications does not hold anymore.

In modern data science applications, the typical setting is: (1) we have a set of operators that cooperate to produce an output (e.g., ML model); and (2) data scientists build pipelines iteratively, and it might take hundreds of iterations to produce a final pipeline. From this setting, we note the following challenges:

One pipeline does not fit all: The assumption of having only one or a fixed number of pipelines that cater to all the users' needs is no longer relevant, i.e., data scientists experiment with multiple pipelines for multiple goals. This makes designing a data cleaning pipeline a significant burden.

Monoethnicity of current data cleaning tools: Data scientists would need to use monolethic tools that were meant to work in isolation of other downstream operators. This makes selecting, and composing the right tools labor-intensive (e.g., how do the systems talk to each other? What type of tuning is needed for each one of them?). As a result, it is typical for data scientists to skip experimenting with existing data cleaning systems, and manually write their own data cleaning routines instead.

Data errors are subjective: In the data cleaning literature, we often find a classification of errors (e.g., outliers). However, we have seen that most errors data scientists run into are ad-hoc and do not fit the traditional classification of data errors. For example, one operator standardizes raw company names into their Legal Entity Identifier representation, but some raw values are left behind.

We have been addressing some of the aforementioned challenges in [2, 3]. And we have ongoing projects that deal with the following directions:

Zero-assumption about data errors : We would like to let the user explore what an error is. In many cases, users do not recognize a data error until they see one. So we need to enable scenario-oriented querying of the input as well as the intermediate data in a data pipeline. For instance, say D_1 and D_5 are datasets produced by two different pipeline operators, an example scenario could be: What happens to the final average of Salary in D_5 if I change an employee's location from NY to MA in D_1?).

Visualization is a requirement, not a luxury: There is almost no research on how to visualize datasets for the purpose of error identification. For example, while working with MGH scientists, we had 350M 2-second segments of electrical brain activity data (EEG). The goal was to train a classifier that predicts, given a 2-second EEG segment, if it is indicative of a seizure. The training data had some noisy segments (e.g., signals whose voltage is too low). Visualizing the signals according to their voltage was key to locate the noisy fragments of the data.

The user does not know databases or SQL: Data scientists come from different technical backgrounds, so assuming they know databsaes or SQL is not a good idea. This is why we are developing new declarative languages for the sole purpose of data debugging[2] to make it easier for users to develop data cleaning pipelines.

Scalability: Running industrial data pipelines is expensive, it took us three days to train the EEG clssifier on a cluster of cloud machines. Since data cleaning often requires quadratic operations, it would be crucial to optimize this process (e.g., through caching) since we know pipeline development is highly iterative.

The talk will feature a rich walk-through of data cleaning scenarios we ran into with our industry collaborators and the key takeaways to inform future data cleaning efforts that can be usable by today's data scientists.

References

[1] I. F. Ilyas and X. Chu. *Data Cleaning*. ACM, New York, NY, USA, 2019.

[2] E. K. Rezig, A. Brahmaroutu, N. Tatbul, M. Ouzzani, N. Tang, S. Madden, and M. Stonebraker. Debugging large-scale data science pipeline using dagger. *Proc. VLDB Endow.*, 2020.

[3] E. K. Rezig, L. Cao, G. Simonini, M. Schoemans, S. Madden, N. Tang, M. Ouzzani, and M. Stonebraker. Dagger: A data (not code) debugger. In *CIDR 2020, 10th Conference on Innovative Data Systems Research, Amsterdam, The Netherlands, January 12-15, 2020, Online Proceedings*. www.cidrdb.org, 2020.

11th Annual Conference on Innovative Data Systems Research (CIDR '21) January 10-13, 2021, Chaminade, USA.