Learning to Restructure Tables Automatically

Joseph M. Hellerstein
UC Berkeley
hellerstein@berkeley.edu

By now, it is widely-accepted folk wisdom that "half of the time in any data analysis project is spent wrangling the data". Analytic algorithms and tools—built on mathematical foundations of matrices and relations—require their data to be lined up in particular rows and columns. In the relational model (known in data science circles as "tidy data"), each row is an independent observation, and each column is a distinct attribute of the phenomenon described by the data. While there are many thorny aspects to data wrangling, perhaps none is more basic than the challenge of getting data reorganized, positionally, into the right form for analysis.

Unfortunately, most data sets do not arrive tidy, in the sense of being relational. In particular, tables made for human consumption—e.g., spreadsheets or web tables—are notorious for requiring reorganization before analysis. Some tables contain a patchwork of rectangular regions, using colors, separator lines, or metadata embedded inside the data to demarcate ranges of cells. Other tables encode data that we want to analyze into the column names (metadata), making those values difficult to query. Some tables simply lay things out awkwardly: data that we want to analyze together (say, a year of daily temperature readings) is spread across multiple columns (e.g., a column per day), making calculations across the full dataset tedious to express.

Every data set is a little different, so reorganizing data into the right rows and columns typically involves generating bespoke, multi-step wrangling *scripts*. Automating this task is fundamentally a problem of program synthesis, usually in a data-centric domain-specific language (DSL).

The *AutoTables* paper by Li, *et al.* in VLDB '23 is an automation breakthrough in this space, setting a new bar for AI assistance in wrangling messy data tables into relational form. The paper is impressive not only for its results, but for its pragmatism and insight.

As the authors point out in framing the problem, a key challenge is the human bottleneck in learning and inference in this domain. Supervised learning is problematic here: human users are not good at reading tables and classifying their problems. At inference time, standard "query by example" UIs are not tenable: it is extremely tedious to write down a "correct" data table in non-trivial scenarios. Instead, the authors attempt to train models and perform inference *without any human input*. How does that work? That is the big "aha moment" at the heart of this very readable paper.

The key idea comes from the algebra: operations in a data wrangling DSL typically have inverses. Hence every operation that makes some data set cleaner also has an inverse that makes the resulting data set dirtier! In particular, if we start with a clean relational table R (say from a relational database), and "dirty it up" with operation O^{-1}, then we have a labeled example: the resulting table $R' = O^{-1}(R)$ is an example of table that can be cleaned via the operation O. Given that simple observation, the rest follows: a training set can be generated on the fly by sampling from the space of (invertible) transformation operators and their parameters, and a space of clean relational tables to be dirtied. Once a model is trained on that data, there is only one query at inference time: "Relationalize This!"

Much of the remaining material in the paper involves featurizing tables so that well-studied learning techniques can capture the patterns associated with R' vs. R in a loss function. The featurization techniques in the paper relate to data visualization, in the sense that they treat data tables as a special case of images. Intuitively, a clean data table is a grid of cells, which—like pixels in an image—have some coherence. The values down a column of a clean table should certainly share certain features. The values of adjacent cells row-wise will exhibit patterns also. The space of expected patterns may be hard for humans to codify, much like the patterns associated with images of cats. But as we know, deep nets are good at learning and discriminating these patterns. The authors cannily borrow tricks used in training computer vision models—e.g., data augmentation—and show that they are effective here as well.

The paper primarily presents its results in the context of coding environments like R and Python. It strikes me that data wrangling automation of this sort may have even more impact in low-code visual tools—perhaps directly in the venerable spreadsheet, the progenitor of low-code innovation! It will be interesting to watch how improvements in automation like this paper are integrated into the user experience of data analysts across industries and application domains. Regardless of how that plays out, this paper is a big step forward on one of the core problems in data wrangling.

The original version of this paper is entitled "Auto-Tables: Synthesizing Multi-Step Transformations to Relationalize Tables without Using Examples" and was published in *PVLDB 16(11)*, July 2023, VLDB Endowment.)