Machine Learning and Big Data: What is Important?

Michael Stonebraker and El Kindi Rezig
Massachusetts Institute of Technology

1 Introduction

At MIT, we have been collaborating on two real-world projects dealing with Machine Leaning (ML) and large amounts of data. The first project deals with performing ML on a 30T data set of EEG (electroencephalogram) sleep study data, in collaboration with Massachusetts General Hospital (MGH). They have collected EEG data on about 2000 patients with time durations varying from a couple of hours to more than 24 hours. In all, they have recorded 21 channels of data, with a total volume of about 30T. Furthermore, they have decomposed this data into about 450M segments, each 2 seconds long. Their project goal is to use ML to classify each 2-second segment "snippet" into one of 12 categories that make sense to doctors. Two papers on this project were presented at the recent VLDB conference [1, 6].

The second project is also in conjunction with MGH and deals with the spread of infections in the hospital. MGH has made 11 years of patient data available, including exactly which room each patient occupied from admission to discharge. For infected patients, they wish to infer how an infection was spread, i.e. by sharing a nurse, by sequentially inhabiting the same room, etc. Because MGH changed patient software in 2016, we first have a data integration problem, which we are addressing using MIT-built ML integration tools [1, 7]. Then, we propose to infer infection pathways from their integrated data.

Finally, one of us has been a principal at an ML data integration company (Tamr) that has done more than 300 ML projects primarily for Fortune 2000 customers. Tamr is in the business of data integration at scale. Typically their projects deal with integrating several-to-many data sets by normalizing the data to a common format, correcting data errors, merging and deduplicating the result, and creating "golden records" for each cluster of records thought to represent the same entity. Typically, input data sets represent parts, suppliers, customers or other entities, and this end-to-end process is usually called "mastering" for a particular entity. The guts of the Tamr system is a collection of ML algorithms to perform schema matching, deduplication and golden record construction.

This paper summarizes our thoughts, based on our experience with these use cases. The remainder of this paper is divided into sections containing our observations.

2 At Scale is Where all the Hard Problems Reside

For example, Toyota Motor Europe (TME) is a Tamr customer. Historically they had country-specific distribution and a country-specific customer database. When a Toyota customer moved countries (say from Spain to France), TME developed amnesia. To correct this situation TME is using Tamr to "master" all European customers. There are 30+ million customer records in 250 databases in 40 different languages. At this scale any N^2 deduplication algorithm is a non-starter, and Tamr has spent a lot of effort on cheaper computations. Obviously, one also needs a parallel multi-core, multi-node execution environment.

Similarly, Glaxo Smith Kline (GSK) is proposing to perform mastering on research data sets often of pharmaceutical compounds. They have many, many data sets with 10+M attributes. Traditional mastering

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

products perform attribute matching by manually drawing lines in a GUI between attributes which represent the same thing. A human obviously cannot draw 10+M lines, and a different (automatic) approach is needed.

In our opinion, if your data set is 1000 records, then any algorithm running "on your wrist watch" will work fine. At scale is where all the demons reside, and traditional solutions fail.

3 Deep Learning is Not for Everyone

So called "deep learning" based on neural networks is the current rage in the research community. However, deep learning is not appropriate for all problems. For example, Tamr customers have very little interest in this technology for two very simple reasons. First, deep learning requires 1-2 orders of magnitude more training data than conventional ML. With structured data in the enterprise, training data required to fit an ML model is always "the high pole in the tent", i.e. there is never enough of it, and getting more is very expensive.

At scale, it is hopeless to obtain training data by asking a human whether given pairs of records are a match or a non-match. That will work fine for 1000 pairs but not 100K pairs. The focus has to be on generating training data using rules or other "weak" methods [8]. For example, if one is mastering enterprise customers, then a rule might specify that two customers with an edit distance between their names below a threshold are probably the same customer. This rule will generate a significant amount of training data, but there will be errors and some sort of validation must be employed. This validation requires skilled enterprise personnel and cannot be done with unskilled contract workers using, say, Mechanical Turk.

For example, the rule in the paragraph above will say that two records for Merck, one with an address in Europe and one in the US, are the same entity. Is this decision correct? Deciding this question can realistically only be entrusted to financial personnel in your enterprise, and these employees are both expensive and busy.

Hence, constructing and validating training data is the most expensive part of the ML process. Any strategy that requires a lot more of it is a non-starter. For the record, German Merck is a different company than US Merck.

The second problem with deep learning is that it is currently an unexplainable "black box". Many Tamr customers want explanations for why a given outcome was selected. If a company is using ML for classifying customers for obtaining credit or other financial reasons, then explanations are necessary. Currently deep learning decisions cannot be explained, and this is an area of active research. Until there is some maturity in this area, conventional ML will be the tool of choice.

In summary, issues with training data and explainability make Tamr customers prefer conventional ML. Perhaps this will change at some point in the future, but for now we expect conventional ML (decision trees, Bayesian methods, etc.) will be the norm for structured enterprise data.

4 Supporting ML at Scale is a Data Management Issue

Obtaining enough training data for many deep learning applications is very expensive. For instance, in our MGH collaboration [6], we built a neural network that required labeling 30M EEG segments for training purposes. Labelling this amount of data manually is infeasible. Hence, the experts (MGH physicians) labeled 8346 EEG segments manually, and then we used clustering techniques to find similar EEG segments from the unlabeled data. This has generated around 37 million EEG segments of training data. However, automatic clustering is error-prone and the training data is also somewhat dirty. Obviously, a good tactic is to iteratively clean the data, train the ML model, and run it, until the results are "good enough". In our demo [1], we showed that we could boost the model's accuracy by 7% just by applying automatic data cleaning tools on the training data.

As a result, improving an ML model can occur in at least three different ways: (1) obtaining more training data; (2) parameter-tuning of the model; and (3) cleaning the source data.

Obviously, one must iterate between these three tactics. In addition, there is a fourth option: data enrichment. The Tamr engineers are often given ML problems where there is simply not enough signal to perform deduplication. Sometimes only the product name is available as a shared attribute, and there is a lot of noise associated with the various ways this gets recorded. Another example is a corporate customer name along with an address. For companies like Staples with many locations, the address has only a small signal value.

In these cases a good tactic is to perform signal enrichment by joining the source tables to other data sources. For example, a successful join of customer name to the Dow Jones Intelligent Identifier (DJII) or the Dow Jones International Securities Number (ISIN) may generate an additional attribute with signal value. Likewise, a join of product name to the Universal Product Code (UPC) may prove helpful. Even if the join can only be performed on a subset of the data, the data network effect (i.e., the value of the added data will benefit all the entities linked or related to it) will cause this to broadly improve results.

Note clearly that enrichment is mostly a data discovery problem, which data management researchers should make a priority research area [4, 5].

Hence, successful ML projects may run hundreds of iterations before they obtain a successful model. Obviously, a good support system is needed to perform model management. Specifically, for each iteration we need to keep track of: the training data, the cleaned data, the state of cleaning, the parameters of the ML model being run, the model's features, the intermediate data, the input and the output data.

One needs a model management system (MMS) to keep track of the "state" noted above, to ensure that everything done during model development can be clearly explained and repeated in production. This is a straight data management problem, and initial work in this area is reported in [9]. A great deal more work is needed in this area, and readers should consider this section as a "call to arms". Moreover, there must be a "human in the loop" to decide what to do next and to back up from failed options when necessary. Optimizing ML platforms for such iteration seems long overdue.

5 ML is not really about ML

A year ago, one of us attended a talk by an iRobot data scientist. (iRobot makes the automatic vacuum cleaners that run around your living room). I asked her how she spent her work week, and here is her answer. "I spend 90% of my time finding data of interest, normalizing it and cleaning it", i.e. data discovery and data integration. "Then, I spend 90% of the remainder fixing my data cleaning errors." In other words, she spends one hour a week or less on data science and 39 hours a week on "data prep". We have talked with many data scientists, and all report similar behavior. We have never met anybody who claimed the answer was less than 80%.

Hence, data scientists are drowning in data prep. Put differently, if you want to help a data scientist do his/her job better, then work on data discovery, data cleaning and data integration. There are many efforts in this area, but much more needs to be done, since this is far and away "the high pole in the tent".

Our favorite complaint is that research institutions appear to spend 90% of their ML effort on algorithms and 10% on data preparation. In our opinion, it should be 90/10 the other way. As real world applications of ML expand, we expect the feedback from enterprises to confirm this. A three star general said exactly this in a recent briefing.

6 ML System Usability

Another issue that often seems overlooked is the usability of ML/data pipeline management systems. Most organizations have their own in-house scripts that perform data preparation and ML. Unfortunately, to benefit from their features, most ML frameworks require users to modify their code to adapt it to those frameworks. We believe this requirement should be reversed: we should strive to build data systems that adapt to users' tools/code and not the other way around. Average organizations often do not have the resources or the manpower to adapt

their tools to specific data frameworks, so they often resort to using their in-house scripts in an ad-hoc fashion. Obviously, frameworks need to become easier for "mere mortals" to use.

Human-in-the-loop data systems have been around for decades. However, we are still a long way from end-to-end data systems that put the human at the forefront [2]. For example, there are few efforts that explore the cognitive cost when recommending a task to a human, i.e., how hard is task X vs. Y. For instance, it is easier to ask a human if two company names refer to the same organization than asking him/her if one company is the parent of another (which requires researching the companies). However, one high-effort answer could lead to greater gains than asking dozens of low-effort questions. This area should be explored to find the best strategies to involve humans and optimize their overall cost.

Lastly, human feedback has been incorporated to solve several point problems in data pipelines; however, it is still unclear how humans should be involved in end-to-end data systems, e.g., from data preparation all the way to analytics. There are many opportunities to incorporate the human feedback at different points in the data processing pipeline, and more research in this area is warranted.

7 Real World Studies Often Deal with Application Specific Data

One of the applications areas where Tamr is used is "oil well mastering". In other words, oil producers like Hess or Exxon have data for wells in various locations. In general, wells are identified by coordinates, but the accuracy of the coordinates as well as the coordinate reference system may have significant uncertainty. As a result, the identifying key for oil wells is imprecise. Therefore, "oil well mastering" requires approximate spatial clustering, to identify records that correspond to the same well. ML in the real world must deal with these kinds of extended types. Many years ago DBMSs began to support type extension, and ML systems need similar techniques. Again, this is a fertile area for extension of ML capabilities.

8 ML/Big Data Papers Should be Required to Use Real Data

ML is sometimes applied to data integration, and often the experimental section justifies the value of the paper's algorithms on "fake" data. In other words, good data is algorithmically perturbed to create incorrect data. It is not surprising that the authors can remove errors that they inserted into a data set!!!

As a community, we should simply reject any paper which uses fake data in its experimental section. Researchers usually argue that real data is not available to them. In our opinion this is a cop out for "I am lazy", and don't want to do the necessary leg work.

9 Debugging ML

ML frameworks should strive to make it easy for users to find out what went wrong with models and to mitigate potential problems. We are still a long way from a full-fledged system that allows users to fully understand what the ML model did and did not do and why. In other words, a full function data debugger would be incredibly valuable, and a first step in this direction appears in [11].

More generally, we need a lot of effort to democratize ML for non-ML-expert users, and to be able to use ML responsibly. It is still very easy to produce garbage results from ML systems! Additionally, humans are not always the best source of training data, as human bias is often transferred to ML models. This will obviously produce biased results. Fairness is emerging as a critical area of research to make sure ML systems are not rigged to unfairly favor one decision over another [3].

10 Conclusions

This paper has presented several areas where we believe current efforts are missing the mark. There should be more efforts devoted to conventional ML a lot more effort to supporting human-driven iteration of an ML pipeline, and a lot more effort on data preparation. Moreover, at scale is where all the demons reside. Data debugging and ML system usability should be research foci. Obviously, researchers in this area should partner with a real-world use case. Otherwise, there is no guarantee that "the rubber will meet the road instead of the sky".

References

[1] El Kindi Rezig, Lei Cao, Michael Stonebraker, Giovanni Simonini, Wenbo Tao, Samuel Madden, Mourad Ouzzani, Nan Tang, Ahmed K. Elmagarmid: Data Civilizer 2.0: A Holistic Framework for Data Preparation and Analytics. PVLDB 12(12): 1954-1957 (2019)

[2] El Kindi Rezig, Mourad Ouzzani, Ahmed K. Elmagarmid, Walid G. Aref, Michael Stonebraker: Towards an End-to-End Human-Centric Data Cleaning Framework. HILDA@SIGMOD 2019: 1:1-1:7.

[3] Babak Salimi, Luke Rodriguez, Bill Howe, Dan Suciu: Interventional Fairness: Causal Database Repair for Algorithmic Fairness. SIGMOD Conference 2019: 793-810

[4] Raul Castro Fernandez, Ziawasch Abedjan, Famien Koko, Gina Yuan, Samuel Madden, Michael Stonebraker: Aurum: A Data Discovery System. ICDE 2018: 1001-1012

[5] Erkang Zhu, Dong Deng, Fatemeh Nargesian, Renée J. Miller: JOSIE: Overlap Set Similarity Search for Finding Joinable Tables in Data Lakes. SIGMOD Conference 2019: 847-864

[6] Lei Cao, Wenbo Tao, Sungtae An, Jing Jin, Yizhou Yan, Xiaoyu Liu, Wendong Ge, Adam Sah, Leilani Battle, Jimeng Sun, Remco Chang, M. Brandon Westover, Samuel Madden, Michael Stonebraker: Smile: A System to Support Machine Learning on EEG Data at Scale. PVLDB 12(12): 2230-2241 (2019)

[7] Dong Deng, Raul Castro Fernandez, Ziawasch Abedjan, Sibo Wang, Michael Stonebraker, Ahmed K. Elmagarmid, Ihab F. Ilyas, Samuel Madden, Mourad Ouzzani, Nan Tang: The Data Civilizer System. CIDR 2017

[8] Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, Christopher Ré: Snorkel: Rapid Training Data Creation with Weak Supervision. PVLDB 11(3): 269-282 (2017)

[9] Manasi Vartak, Joana M. F. da Trindade, Samuel Madden, Matei Zaharia: MISTIQUE: A System to Store and Query Model Intermediates for Model Diagnosis. SIGMOD Conference 2018: 1285-1300

[10] Tim Kraska:Northstar: An Interactive Data Science System. PVLDB 11(12): 2150-2164 (2018)

[11] El Kindi Rezig, Lei Cao, Giovanni Simonini, Maxime Schoemans, Samuel Madden; Nan Tang; Mourad Ouzzani; Michael Stonebraker: Dagger: A Data (not code) Debugger. CIDR (2020)