Research Statement
Jiannan Wang

With the increasing amount of available data, turning raw data into actionable information is a requirement in every field. However, one bottleneck that impedes the process is data cleaning. Data analysts usually spend half of their time cleaning data that is dirty — inconsistent, inaccurate, missing, and so on — before they even begin to do any real analysis. It is a time-consuming and costly process but necessary for obtaining high-quality answers from dirty data. This is further exacerbated in emerging Big Data scenarios when data volumes are increasing, or when the data is integrated from a larger variety of sources.

In my research, I aim to design algorithms and develop systems to obtain timely, high-quality answers from large, dirty datasets. Working primarily from a Database Systems perspective, I study three fundamental ways of dealing with errors in dirty data:

1. Data Cleaning [1-12]: Detecting and removing errors from data
2. Error-Tolerant Search [13-17]: Tolerating dirty data without data cleaning
3. Approximate Query Processing [19,20]: Efficiently assessing and correcting errors in large data sets using sampling

My PhD research focused on the first two points. I developed algorithmic data-cleaning techniques and built hybrid human-machine cleaning systems to help people to clean data more efficiently and more reliably. I proposed an interactive and error-tolerant search paradigm that can help people to explore dirty data efficiently.

My postdoc research has focused mainly on approximate query processing. I proposed a sample-and-clean framework that applied statistical methods to estimate query answers on dirty data along with confidence intervals. I also began a line of research to expand the use of data cleaning techniques and insights to areas that are not traditionally considered data cleaning problems such as materialized view management.

My research approach typically follows a common pattern of integrating systems building and theoretical reasoning. I first identify research problems from the challenges faced by real-world systems and formalize these problems in theory. I then use the theory to guide the development of the systems, and finally validate the theory using the developed systems. I truly believe the best way to validate research ideas is to build real systems and evaluate them in practice. For example, my work in interactive and fuzzy search has been productized by a startup company (SRCH2), and my data cleaning algorithms are available to download publicly.

Data Cleaning

To obtain timely and high-quality answers from dirty data, the first direction that I have explored is to develop efficient and reliable data-cleaning techniques and systems. My focus is not only on algorithmic data cleaning techniques, but also on hybrid human-machine cleaning systems.

Algorithmic Data-Cleaning Techniques

The algorithmic data-cleaning techniques that I develop have the goals of being fast and easy-to-use, which make these techniques applicable to a wide range of domains and scale sizes.

Fast: Data cleaning algorithms are usually expensive to run. If I can design faster ones, it will save people time. To this end, I investigate some basic data-cleaning operators, and study how to implement them efficiently. I propose Trie-Join [8, 12], a trie-based framework for edit-distance based similarity join. This operator is widely used to capture spelling errors in databases (e.g., “Berkeley” v.s. “Berkerley”). Trie-Join performs very well for short strings. I also propose Pass-Join [9], an algorithm that is particularly optimized for processing long strings. To detect more types of data errors, I propose a new similarity function and develop efficient algorithms to support similarity joins on this function [3, 11]. I participated in the String Similarity Search/Join Competition in EDBT 2013\(^2\). My similarity-join algorithms won first place, and ran 10–100x faster than the second best team.

Easy-to-use: Parameter tuning is always tedious work for data cleaning. Different settings of parameters may lead to widely varying results. I study how to automate the process of parameter tuning in order to produce more reliable cleaning results [10]. Inspired by supervised machine learning, I replace the work of tuning parameters with the work of labeling a set of training examples, by designing optimization algorithms to select suitable parameters based on the training examples. This paradigm shift makes parameter tuning much easier than before since people can easily label a few examples rather than tune parameters directly. In addition to the accuracy of cleaning results, I find that the efficiency of many cleaning algorithms is highly dependent on parameter settings. I build a cost model to estimate the cost of running a cleaning algorithm under different parameters, and devise an adaptive framework to automatically select the best parameters that lead to the minimum cost [7]. Experimental results show that the cleaning algorithms that are developed based on our adaptive framework are much more efficient than those based on the state-of-the-art framework.

\(^1\) [http://www.srch2.com](http://www.srch2.com)
\(^2\) [http://www2.informatik.hu-berlin.de/~wandelt/searchjoincompetition2013/Results.html](http://www2.informatik.hu-berlin.de/~wandelt/searchjoincompetition2013/Results.html)
Hybrid Human-Machine Data Cleaning Systems

Algorithmic data-cleaning techniques have been improving in quality, but remain far from perfect. Crowdsourcing platforms offer a way to bring human insight into the process. A key question I investigate in my work is how to build hybrid human-machine data cleaning systems that can reduce cleaning cost while providing good answer quality.

Entity Resolution: Entity resolution (ER) in database systems is the task of finding different records that refer to the same entity. It is central to data cleaning and data integration. A machine-only ER approach often falls short on quality, and a human-only ER approach is too slow and expensive. I propose a hybrid human-machine approach in which machines perform an initial, coarse pass over all the data and people verify the most likely matching pairs [6]. I show that for such a hybrid system, generating the minimum number of verification tasks of a given size is NP-Hard, and develop a novel two-tiered heuristic approach for creating batched tasks. I identify the importance of exploiting transitivity to reduce the cost, and present a new framework for implementing this technique [4]. Based on these ideas, I built CrowdER, a hybrid human-machine ER system. This system shows that a hybrid human-machine data cleaning system can achieve both good efficiency and high accuracy compared to machine-only or human-only alternatives.

Data Repairing: Data repairing is the task of correcting erroneous attribute values in a database. A typical automated data repairing procedure is to first define some integrity constraints (e.g., functional dependencies) for the database, then detect attribute values that violate the integrity constraints, and finally, to search for possible corrections to the violations. As with entity resolution, achieving high quality in automated data repairing is difficult. When using humans to repair data, it becomes much more time-consuming and costly. I propose fixing rules, a new class of cleaning rules to solve this problem [1]. Fixing rules can be first learned from people and then applied to repair the data automatically. I study fundamental problems (consistency, implication, and determinism) associated with fixing rules, and establish their complexity. I show that fixing rules can perform more reliable data repairing than existing automated repairing approaches, while requiring lower cost than user-guided data repairing methods.

Approximate Query Processing

As data volumes continue to grow, data cleaning becomes increasingly time consuming. This problem motivates me to investigate the second, novel way of dealing with data error. That is, cleaning a small sample of data and then using the cleaned sample to estimate query results on the full dirty data. In my most recent work, I have developed SampleClean and SVC, two research projects based on this idea.

SampleClean: Fast and Accurate Query Processing on Dirty Data

I joined AMPLab at UC Berkeley as a Postdoc in 2014, where I started the SampleClean project. AMPLab is a collaboration of Systems and Machine Learning researchers whose vision is to integrate Algorithms (Machine Learning), Machines (Cloud Computing) and People (Crowdsourcing) to make sense of Big Data. To integrate Algorithms and Machines, the lab developed a variety of open-source software, such as Spark and MLBase. These projects, however, largely ignored the “People” aspect of the project. To integrate People, we are developing the SampleClean system, a new open-source software component in the BADS (Berkeley Data Analytics Stack), which allows users to obtain accurate query results from dirty data, at significantly reduced time and cost. I currently lead a team of two PhD students and two undergraduates who are developing the system. An initial version of the system was demonstrated recently at AMPCamp 5.

The key idea of SampleClean is to apply data cleaning to only a small sample of the data and then use the results of the cleaning process to lessen the impact of dirty data on query answers [19]. Central to the SampleClean system are two main components. (1) Asynchronous Data Cleaning Pipelines. Users can construct a data-cleaning pipeline using (machine-only, human-only, or hybrid human-machine) data cleaning operators implemented by the SampleClean system, and apply the pipeline to clean the sample data asynchronously. The asynchrony feature avoids blocking users by time-consuming data cleaning, and enables them to see intermediate query results during the cleaning process. (2) Approximate Query Processing. Since the data-cleaning pipeline is only applied to the sample data, the question is how to leverage the cleaned sample to answer users’ queries on the entire dirty dataset. We propose two query processing methods in SampleClean. One is to estimate the query results only based on the cleaned sample. The other is to use the cleaned sample to correct the errors in query results over the dirty data. Both of the methods are challenging since data errors change the sampling statistics (e.g., duplicate records are more likely to be sampled). SampleClean addresses those statistical challenges, and has been shown to return unbiased query results with analytical confidence intervals.

SVC: Stale Materialized View Management

Materialized views (MV), stored pre-computed query results, are widely used to facilitate fast queries on large datasets. During the last 30 years, the research community has thoroughly studied MVs, and all major database vendors have added support for them. In a world of ever-increasing data sizes, MVs are becoming even more important, both for traditional query processing and for more advanced analytics based on linear algebra and machine learning. However, a big challenge of MVs is

5 http://ampcamp.berkeley.edu/5/
that they can become stale when the underlying data is changed. In the SVC project, we model staleness as a type of data error and treat the view maintenance problem as a data-cleaning problem [20].

My main contribution in SVC is to develop efficient sampling techniques for different types of MVs. Unlike existing MV sample problems, SVC needs to sample from the updated records in a fresh MV. This problem is challenging because a simple sampling technique can be as expensive as full view maintenance. I proposed a hash-based sampling framework and studied how to modify hash keys according to various relational algebra operators. SVC complements existing deferred maintenance approaches. When the MVs are stale between maintenance cycles, we can apply SVC for approximate results for a far lower cost than having to maintain the entire view. Our analysis and experimental results suggest that SVC is significantly faster than full view maintenance, and also can give highly accurate results for a variety of queries and views.

**Error-Tolerant Search**

In addition to data cleaning and sampling, error tolerance is another way of dealing with data errors. I applied this idea to the Keyword Search research. Keyword search is a popular way to query and explore textual data. However, when underlying data is dirty or search queries contain typos, keyword search may return unsatisfactory results. In this situation, users need to modify or refine their search queries multiple times in order to reach the desired information. To overcome this limitation, together with researchers at UCI, we propose a new search paradigm [14, 16], which extended traditional keyword search with two new features: (i) Error-tolerance: tolerating errors in queries and data; and (ii) Interactivity: providing instant feedback to a query letter by letter. This type of search-as-you-type paradigm allows users to find results “on the fly” and enables them to dynamically modify or refine queries, removing the major barrier between queries and search results.

The new search paradigm poses significant computational challenges, since it requires both a high interactive speed and the capability of relaxing keyword conditions. To achieve an interactive speed, the search response time should not exceed milliseconds (typically within 100 ms). I address these computational challenges by employing novel index structures, caching techniques, search algorithms and parallel computing [13]. I developed two real prototypes using these techniques: (i) iFSearch is a Firefox extension that adds interactive and error-tolerant search capabilities to the Firefox address bar [15]; (ii) iPubmed is a new Pubmed search system (jointly developed by me and a PhD student at UCI) that enables biomedical scientists to explore millions of MEDLINE publication records interactively [17].

**Future Research**

**Machine Learning on Dirty Data**

In the SampleClean project, I develop new techniques to enable fast and accurate query processing on dirty data. The techniques work very well for typical aggregation queries (e.g., SUM, COUNT, and AVG). In modern data analysis systems, however, supporting machine-learning algorithms to enable more sophisticated data analysis becomes increasingly important. Therefore, a challenging research problem is how to efficiently train a high-quality machine-learning model on dirty data. Currently, I am working with people, who have machine-learning background at AMPlab, to define the scope of the problem. I plan to study the problem by considering three sources of data errors.

- **Staleness.** One of key characteristics of Big Data is velocity, which means that new records arrive at an increasingly fast rate. Maintaining an up-to-date machine-learning model in this scenario is not an easy task. Current systems often solve this problem by maintaining the model periodically. But between maintenance periods, the model becomes increasingly stale. I believe there is a better trade-off between no maintenance and full maintenance. I would like to leverage the idea in the SVC project and investigate how to learn a delta model to compensate for the influence of data updates on an old model. I envision that this new approach can offer significant improvement in time over training a new model and significant improvement in accuracy over keeping an old model.

- **Data Integration.** Another key characteristic of Big Data is variety, which means that data is often integrated from different data sources. Data integration is an expensive and error-prone process. When training a machine-learning model on integrated datasets, the model can be problematic. I plan to solve this problem by looking at data integration and machine learning at a single process, and investigating how they can affect each other. This idea leads to many interesting research questions. For example, how much integration work needs to be done before getting a satisfactory model? How can we optimize data integration pipelines such that data errors have less effect on a model?

- **Data Collection.** Real-world datasets are often collected from unreliable sources, such as sensor networks or Internet users. Training a good machine-learning model on such datasets either requires high-quality data-cleaning algorithms (to remove dirty data values upfront) or requires robust machine-learning algorithms (to handle dirty values automatically). I would like to investigate another idea, similar to CAPTCHA, to stop dirty data from being input at the data collection stage. Since our goal is to use the collected data to train a machine-learning model, interesting research problems include: What is the best data-collection strategy for machine learning? How can we reduce the cost of data collection while still training a good model?

---

Repeatability of Scientific Data Processing Platform

With the rise of big data, there is a paradigm shift happening in every scientific field, namely from experimental, theoretical or computational science to data-driven science\(^6\). In the new paradigm, scientists typically need to create a complex data-processing pipeline (including data integration, data cleaning, data sampling, etc.) for turning massive datasets (captured by instruments or generated by simulations) into scientific discoveries. At the same time, the repeatability of scientific discoveries is vital to every scientific field. When scientists make a scientific discovery, they would like to ensure that others can validate their results by repeating the process of the discovery. In view of the popularity of data-driven science and the importance of repeatability, I would like to explore how to build a new data processing platform for scientists to facilitate the repeatability of scientific discoveries.

To fulfill this vision, there are many challenging problems that need to be addressed. For example, (1) during the data processing, some parts of data can be removed and some can be modified. In order to repeat the process, a system has to keep track of how data is changed in each procedure. This can be difficult, especially when data is very large or data is changing frequently. (2) Data-driven scientific discovery often requires processing a large amount of data. In order to repeat the discovery, re-running the entire data-processing pipeline can be very costly and time consuming. One possible solution is to cache some intermediate results to avoid recomputation. But how can we guarantee that the cached intermediate results are the same as the ones obtained by recomputation. Thus, there is an interesting trade-off between efficiency and confidence in repeatability processes.

I have many years of experience in building data processing systems, but this experience has been mainly from the computer science perspective. To address the above challenges and build a repeatable scientific data processing platform, I plan to collaborate with scientists from other fields and gain a deep understanding of their domain-specific workflows, datasets, and tools. My postdoc work on data cleaning for automation systems is an initial attempt towards this kind of collaboration \([18]\), where I worked with researchers at the Berkeley Automation Science Lab, and helped them clean sensor data to train a better control model. In my future research, I plan to extend this line of research, collaborating with scientists from different fields to identify fundamental challenges and build a data processing platform that enhances repeatability of scientific discoveries.

References

[16] Guoliang Li, Shengyue Ji, Chen Li, Jiannan Wang, Jianhua Feng. Efficient Fuzzy Type-Ahead Search in TASTIER. ICDE 2010:1105-1108. (Demo)

---

\(^6\) http://research.microsoft.com/en-us/collaboration/fourthparadigm/