Human-in-the-Loop Data Analysis: A Personal Perspective

AnHai Doan
University of Wisconsin-Madison

ABSTRACT

In the past few years human-in-the-loop data analysis (HILDA) has received significant growing attention. Most HILDA works have focused on concrete problems. In this paper I take a step back and discuss several “big picture” questions regarding HILDA. First, I discuss problems that I believe should fall under the scope of the field, including some that have received little attention, such as fostering user communities that develop data repositories and tools. Next, I discuss important aspects in developing HILDA solutions that I believe should receive more attention. These include solving problems that real users care about, developing how-to guides to users, building end-to-end systems (such as extending the “Pandas system”), developing challenges and benchmarks, and developing a theory of human data interaction. Finally, I speculate about the future of the field, and discuss the dangers it can face, given that many other communities are also working on related problems. I argue that a focus on end-to-end problems and system building is important for us to thrive and make significant impacts.

1 INTRODUCTION

Over the past decade there has been a growing realization that many data management tasks either require or can significantly benefit from the involvement of humans [1]. As a result, this emerging topic has received significant attention, in the newly created series of workshops and elsewhere, and has been referred to as *human-in-the-loop data analysis (HILDA)*.

Most current HILDA works however have focused only on specific problems (e.g., visualizing big data, cleaning, entity matching). Going forward, I think we should devote more attention to examining “big picture” questions, such as “What is the scope of HILDA?”, “What are the important aspects that we should pay more attention to?”, “Where is the field going?”, and “What should we do to stay relevant and make major impacts?”. Exploring these questions help us understand the broader context of what we are doing, better evaluate our work, discover neglected topics, adjust our directions if necessary, connect to other fields, and more. More importantly, it can help us understand if there is enough here to build a solid and coherent foundation for a new field, or if this is just a loose set of techniques.

In this paper I take the first step in exploring some of these “big picture” questions. First, I discuss the types of data problems I believe the field should address and the types and roles of humans involved. I argue that while so far the HILDA community has rightly focused on problems that extract insights from data (e.g., data wrangling and analysis) or build knowledge bases/graphs, it should pay more attention to the increasingly popular problems of fostering user communities that develop data repositories and tools. For example, many domain sciences are building many data repositories and the communities surrounding those, and many communities have also sprung up to develop data tools (such as the R or Python communities). They can certainly benefit from the data management expertise of the HILDA research community.

Second, I discuss important aspects in developing HILDA solutions that I believe should receive more attention. These include solving problems that real users care about, developing how-to guides to users, building end-to-end systems, developing challenges and benchmarks, and developing a theory of human data interaction. In particular, I argue that *there is already an existing HILDA system out there, which is very popular and growing rapidly.* This “system” is the ecosystem of the Pandas Python package and associated packages such as matplotlib, scikit-learn, pandas-profiling, etc. I argue that rather than building isolated HILDA systems, the HILDA community should consider *“extending this Pandas system”, by developing more Python packages that solve HILDA problems that arise for the users of this system.* This can bring numerous benefits and maximize our impacts. I also argue that while we have isolated observations about how humans interact with data, we lack a coherent and comprehensive theory of human data interaction (HDI) and that such a theory should be developed, to provide a solid foundation for the field.

Finally, I speculate about the future of the field, and discuss the dangers it can face, given that many other communities (e.g., AI, machine learning, HCI, visualization, KDD, and more [7]) are also working on related problems. These dangers include a loss of identity, playing catchup, and a lack of impact. I argue that a focus on end-to-end problems and end-to-end system building is important for us to thrive and make significant impacts.

As described, this paper is *not* a survey. It merely provides a personal perspective, and is not meant to be comprehensive. Rather, it is meant to provoke more discussion and reflection on the “big picture” questions about this important emerging field.

2 DEFINING THE PROBLEMS

HILDA has typically been defined as *studying data problems that either require or can significantly benefit from the involvement of humans.* For example, data exploration requires human involvement by definition. Entity matching does not have to. It can employ just algorithmic solutions, but can achieve significantly higher accuracy if these solutions can involve humans [8, 20].

The above definition is reasonable but too general. To obtain a better understanding about the scope of the field, and to foster a discussion about this scope, in this section I will elaborate on the

HILDA'18, June 10, 2018, Houston, TX, USA

ACM ISBN 978-1-4503-5827-9/18/06…$15.00
https://doi.org/10.1145/3209900.3209913

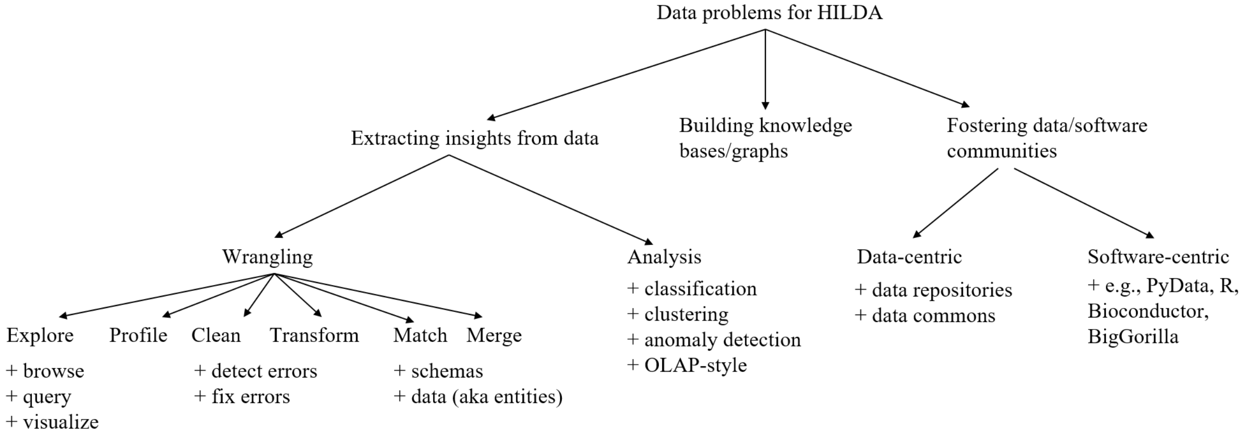

Figure 1: A taxonomy of data problems that require or can significantly benefit from human involvement.

above definition, focusing on the key notions of "data problems" and "humans".

Data Problems: Figure 1 describes a taxonomy of data problems that I believe HILDA should consider. At the highest level, this taxonomy lists three subtrees: "Extracting insights from data", "Building knowledge bases/graphs", and "Fostering data/software communitites". As we move from the left to right on this figure, humans play increasingly more roles and more complex roles in the data management process.

(a) Extracting Insights from Data: This subtree (shown in the left side of Figure 1) should be familiar to most database researchers, and the vast majority of HILDA work has focused on problems here (presumably this is where the phrase "data analysis" in the name HILDA comes from). This subtree lists the main problems that must be solved to go from raw data to insights [1]. Specifically, to obtain insights, we often must perform *data wrangling* (a.k.a. data preparation and integration, data curation, data munging, etc.) to clean and combine the data from disparate sources into a single clean unified dataset, and then perform *data analysis* on this dataset. Examples of work on data wrangling include [17] and companies such as Trifacta and Tamr.

Data wrangling often requires data exploration, profiling, cleaning, transformation, matching, and merging [1, 11]. Data exploration in turn often requires browsing, structured/keyword querying, and visualization, and so on, as shown in Figure 1.

(b) Building Knowledge Bases/Graphs: In recent years, the problem of building knowledge bases/graphs (see the middle part of Figure 1) has also received significant attention [1]. Solving this problem often requires solving those listed under "Wrangling" and "Analysis". As a result, this problem is also generally considered to fall under HILDA's scope.

(c) Fostering Data/Software Communitites: In contrast to the above two parts, the last part of Figure 1, the subtree "Fostering data and software communities", is still relatively unknown to database researchers and thus remains most intriguing. It describes the problems of growing data and software-centric communities (where "software" here refer to tools and systems that manage a non-trivial amount of data).

Numerous such communities have been developed in the past few decades. Early data-centric communities targeted the broader public. One of the earliest such communities (dating back to 2001) is Wikipedia, where tens of thousands of volunteers work together to build a large data repository consisting of both text and structured data (e.g., infoboxes). Another prominent early data-centric community is Facebook (dating back to 2004), where billions of people "build" a giant relationship graph and share data.

More recently, domain scientists have also created many data-centric communities, each of which maintains one or more *data repositories*, where scientists can submit, curate, and consume data. In fact, this is now a very prominent trend in domain sciences. As a particular domain science X becomes increasingly data-driven, a set of enterprising scientists in X will band together to set up a data repository, where scientists worldwide can submit their data, use data already in the repository, and help curate the data. These enterprising scientists often run annual workshops to educate fellow scientists in X on how to use and contribute to the data repository.

Examples include the Environmental Data Initiative at *environmentaldatainitiative.org* for the environmental science community, the UMETRICS initiative at *btaa.org/research/umetrics* for the science policy community, and many more. Among the domain sciences, perhaps the biomedicine community has made the most progress. They have been advocating and implementing the notion of *data commons* (see *https://commonfund.nih.gov/bd2k/commons*), which are communities of users, datasets, tools, and infrastructures, all working together to generate, curate, disseminate, and consume data. (As such, building a data-centric community is clearly very different from the problem of building a knowledge base.)

In addition to data-centric communities, where the focus is to collaboratively build, curate, and use datasets, in the past two decades many software-centric communities have also been developed, where the focus is to collaboratively build, curate, and use *open-source tools and systems* that process data. Perhaps the most prominent examples are the R and Python ecosystems of data science tools. But many smaller tool ecosystems exist, such as Bioconductor (a thriving community of R tools for genomic data, see *bioconductor.org*). Another example is BigGorilla, a recent attempt to foster a community of data preparation and integration tools as a part of the Python ecosystem, see *biggorilla.org* [24].

As domain sciences become increasingly data driven, and data science becomes increasingly pervasive, the trend of building data and software-centric communities will only accelerate. As a result, I believe that the database community should devote more effort to these important problems (they are clearly falling under the purview of data management, after all). There have been some initial efforts (e.g., [4, 9, 14, 24]), but much more is necessary. In particular, the HILDA community is in a good position to study these problems, as they clearly involve humans as first-class citizens. These problems will require solutions to those listed under "Extracting insights from data" and "Building knowledge bases/graphs", but will raise many new challenges, including how to organize the humans to accomplish certain goals, how to design incentives for community members to contribute, and how to communicate effectively among the members.

Humans: I propose to categorize the humans involved in HILDA problems along three dimentions: size of the human group involved in the problem, technical sophistication, and roles in the data management process.

(a) Size: We can distinguish (1) a single user, (2) a set of collaborators, e.g., a team of domain scientists working on a problem, (3) a crowd of workers, where the workers all do similar tasks, such as Mechanical Turk workers answering yes/no questions, (4) a crowd of workers, where the workers may do different tasks and coordination is often necessary, and (5) a community of users, e.g., to build a data or software-centric community).

(b) Technical Sophistication: We can distinguish (1) lay users, who have no coding skills, (2) intermediate users, who have limited coding skills, e.g., a domain scientist, and (3) power users, e.g., data scientists, programmers.

(c) Role in the Data Management Process: We can distinguish (1) creator of data, (2) curator of data, (3) consumer of data, and (4) community members [1].

A Large HILDA Problem Space: Combining the above three human dimensions with the taxonomy of data problems in Figure 1 helps us define a large problem space for HILDA. For example, a problem is this space is data profiling where a single lay user wants to ask a structured query over a table. Another problem is data cleaning, where a group of domain scientists want to collaboratively clean a dataset, etc.

Clearly, the HILDA community has addressed many problems in this space, but has paid little attention to many others, such as those involving building data/tool communities. Another important problem that has received little attention involves a set of collaborators, such as a group of domain scientists spread over multiple locations wanting to work together on a HILDA problem. My experience suggests that this problem is pervasive. Addressing it raises difficult challenges, such as building cloud-hosted scalable solutions so that users in disparate locations can collaborate, and developing methods to enable effective communications among the users (e.g., when collaboratively labeling a dataset, how can the users quickly converge to the same labeling definition, such as the same definition of match in entity matching? [10]).

3 DEVELOPING THE SOLUTIONS

I now discuss important aspects in developing HILDA solutions for the above problem space that I believe our community should devote more attention to.

Solve Problems that Real Users Care About: Many HILDA works focus on developing algorithmic solutions to *carefully defined abstract problems*. For example, given two tables A and B, match them (i.e., find tuple pairs across A and B that refer to the same real-world entity) with as high accuracy (and as little cost) as possible. As another example, given a table A, clean it by detecting errors (e.g., outliers and incorrect values) and repairing them.

In practice, *real users often do not know what to do with the solutions/tools being offered for these abstract problems.* For example, in entity matching (EM), it is rarely the case that a user just wants to match two tables. Often he or she wants to match them with a desired target accuracy, or wants to match better than an existing EM tool already in deployment, and so on. This raises additional problems, such as how to estimate the matching accuracy of a tool, and what to do if the given tool cannot reach the desired accuracy (in this case the user may also want to explore the tables to obtain a better understanding, to clean the tables, to debug the tool, etc.).

As another example, it is rarely the case that a user wants to clean the whole table, because real-world tables are often huge and cleaning them in their entirety is just too expensive. Instead, the user may just want to clean as little as possible for a downstream application (e.g., EM, or to answer a research hypothesis). In both examples, *solutions to today abstract problems offer relatively little help* (e.g., how to estimate EM accuracy? what if using the solution the user still cannot reach the desired accuracy?). As a result, real users are often reluctant to use such solutions, especially if that means having to spend a non-trivial amount of time installing and learning how to use the tools.

Worse, once the abstract problem has been set up (and some solutions have been published), a race is on to develop ever-more-complex technical solutions to squeeze out more accuracy gain, such as developing a highly sophisticated crowdsourcing solution to maximize EM accuracy. The danger of this is that such a solution may not even be necessary in most real-world settings. For example, if a user wants 90% EM accuracy, it may be the case that after cleaning the two tables A and B a little bit, a relatively simple EM algorithm (e.g., a classifier trained on a set of labeled tuple pairs) can already reach that desired accuracy. *So our work ends up further and further removed from actually being helpful to real users.*

To address these problems, I believe we need to work more on real-world HILDA scenarios, with real users. *A highly promising way to do so for academic researchers is to talk with domain scientists at the same university.* Twenty years ago this might have been difficult, because most of the data was still at companies. But the situation has dramatically changed. At virtually any university now, often within a walking distance from the CS department, there are many domain science groups that are awash in data, where the scientists have many pressing HILDA problems to solve. My experience suggests that it may take a while to talk and find several groups with which one can start an effective collaboration. But if persistent, one is likely to find such groups. *Trying to work with and solve their problems is often an eye-opening experience, which can quickly drive home where the real "pain points" are, and thus which problems should be addressed.* Another promising solution is for the HILDA community to identify and publish a set of challenge problems and benchmarks, as I will discuss soon below.

Do Not Underestimate the Need for a Human Loop: It is often said that solving HILDA problems requires some humans "in the loop". The image this conveys is that to solve a HILDA problem, we will employ an (algorithmic) tool, which will try to automatically solve the problem as much as possible, but will consult the humans where necessary. I refer to this as the "tool loop" (with humans being part of it).

In practice, there are indeed many tools like this. But just focusing on developing them is far from sufficient. Recall that we discussed earlier the need to focus more on HILDA problems that real users care about. Such problems are often referred to as "end-to-end" problems. While subjective, the notion "end-to-end" conveys the idea that in such problems we start on one end with the raw data, and finish at the other end with something that the real user cares about (e.g., an EM result with a desired accuracy). When solving such end-to-end problems, it is often the case that there is no single tool that can automate the entire process, or even a large part of it (at least with today technologies).

Instead, what is more likely is that there is a plethora of tools, each solving a part of the problem. The user must still somehow know what to do, step by step, from "one end" to "the other", and where to use which tool. I refer to this as the "human loop" (with tools being part of it).

It is critically important that we develop guidance for real users to execute this human loop. Otherwise they have no idea what to do. (Again, some time spent with domain scientists trying to solve their problems will very quickly bring home this point.) Such guidance is *not* user manuals for how to use the tools. Rather, it is a step-by-step instruction to the user on how to start, when to use which tools, and when to do what manually. Put differently, it is an (often complex) *algorithm* for the human user to follow. There have been some recent efforts at developing such guides, called "how-to guides", for EM [20]. But significantly more effort is needed here.

We Must Build Systems or Leverage Existing Ones: It is obvious that to maximize our impact, it is not sufficient to just develop algorithmic solutions, we need to also build systems that can be deployed to help real users and to evaluate our solutions.

These systems typically fall into two groups: *on-premise* and *cloud-hosted*. On-premise systems are those that a single user may download, install, and run (either on a single machine or on a cluster of machines). Cloud-hosted systems typically provide a Web-based interface for users. They can be public (i.e., anyone can use) or privately deployed (e.g., within a single company, to handle sensitive data).

In practice, we will need both kinds of system. On-premise systems are easy for a single user to download and use, but are ill suited for collaboration (e.g., among several domain scientists spread over multiple locations). Cloud-hosted systems are good for collaboration, but may have fewer capabilities and require more work in hosting and maintaining.

Currently the HILDA community has done only limited work on both kinds of systems. As far as I can tell, there have been very few cloud-hosted systems being built and deployed (e.g., [13, 15] and Trifacta being deployed on Google Cloud Platforms). The situation looks better for on-premise software. Here many systems have been developed. But there are several problems. First, many of them are often just research prototypes, which are hard to use and are often not powerful enough for real-world applications. Second, they are isolated systems each following a particular architecture/API. This makes it hard to combine, to make "the whole greater than the sum of the parts". *Worse, since there is no agreed-upon single system that the whole community works on, it is hard to get a sense on whether we are making progress at all.*

To address this problem, I argue that *there is already an on-premise HILDA system out there, which is very popular and growing rapidly.* This system is the Pandas Python package and associated packages, e.g., matplotlib, scikit-learn, pandas-profiling, etc. Together they form a powerful ecosystem of Python packages, a "system" indeed, but not one in the traditional sense of stand-alone monolithic systems such as RDBMSs. This "system" is already being used by numerous users (e.g., domain scientists) to explore, profile, clean, transform, and perform analysis on the data, in short to solve all those problems listed under the subtree "Extract insights from data" in Figure 1. This "system" is also growing rapidly: many new Python packages are being created and published daily to work on Pandas dataframes.

I propose that, instead of building isolated systems or new systems from scratch, the HILDA community consider joinning forces to work on improving this "Pandas system", by developing new Python packages that solve HILDA problems that arise for users of this system. This can bring many significant benefits:

- First, by building on this "Pandas system", we can instantly leverage many capabilities, and thus avoid building weak toy research prototypes.
- Second, many domain scientists work with this "system". So by extending it, we have a better chance of convincing these scientists to use our tools.
- Third, we can teach the "Pandas system" and our tools in a seamless fashion in our classes, thereby educating our students in practical data science tools (that they should know before graduating) yet obtaining an evaluation of how our tools work.
- Fourth, for our students, developing (and maintaining) Python packages are much easier compared to developing complex stand-alone systems (such as RDBMSs). So academic researchers stand a better chance of developing "systems components" that can be used in practice.
- Fifth, tools developed by different HILDA groups have a better chance of being able to work together, thereby "making the whole greater than the sum".
- Sixth, as it currently stands, the "Pandas system", while being very popular, is actually very weak in terms of solving HILDA

problems in exploring, profiling, cleaning, transformation, etc. So this is a real opportunity for the HILDA community to contribute.

- Finally, if the whole community focuses on a single system, we have a better chance of measuring whether we are making good progress.

Building on our work in the "Pandas system", we can then work on cloud-hosted systems, which (unlike Pandas data frames) can handle data that is larger than memory and can execute tasks in a distributed/parallel fashion, among others. Again, there has been some work on this in the PyData community, but they are still preliminary, and can significantly benefit from the data management expertise of the HILDA community.

Need Benchmarks and Challenges: In addition to working on a single system, as proposed above, working on a set of benchmarks and challenges also helps focus the community's effort. Consequently, I believe developing benchmarks and challenges is critical.

Recent work has explored developing benchmarks [3, 12]. But these benchmarks are only intended for data exploration and visualization. We need benchmarks also for other HILDA problems, such as data cleaning, entity matching, profiling, etc. Further, benchmarks for HILDA problems are notoriously difficult to develop, because they need to somehow account for user actions [12]. Finally, benchmarks alone are not enough, because it may still encourage the development of ever-more-complex algorithms on isolated problems, which are not very useful in practice.

To address these issues, we should also develop end-to-end challenge problems, which provide real-world data and pose real-world tasks that real-world users care about. For example, *it may be possible to select a few domain sciences where there have been efforts to build data repositories, and then pose those problems as challenge problems for the HILDA community.* For this to work, the domain sciences must be such that (a) the data is understandable for most HILDA researchers (so biomedicine may not be appropriate), (b) the data is not sensitive, and (c) there are domain scientists who are willing to work with the HILDA community. For example, a possible challenge problem may be helping environmental scientists explore, profile, clean, and transform 42,700+ CSV tables in the data repository *https://portal.edirepository.org/nis/home.jsp.* It may then be possible to capture both the data and the tasks for some challenge problems at some point in time, together with detailed documentation, and then post those as benchmarks. Obviously far more details need to be worked out, but for the HILDA community to make rapid progress, we need benchmarks and challenges, the same way that many benchmarks and challenge problems have helped the RDBMS community focus and rapidly develop relational data management systems during the 1970s and 1980s.

Need a Theory of Human Data Interaction: To work effectively with humans (in the loop), we need to understand them thoroughly. What is easy/difficult for them in terms of data processing? What are their biases, weaknesses, strengths, preferences, etc. regarding data? There are many "rules of thumbs" and observations that have been mentioned in the literature. I believe we should codify them, examine, and develop a coherent "theory of human data interaction (HDI)".

It is important to note that HCI and visualization researchers have pioneered a large body of work behind HDI (such as [2, 5, 18, 22, 23]). Recent work in the HILDA community has also started examining the limitation of human perception and perception-driven database optimizations [19, 21, 26], exploiting observations such as "users have limitations beyond which he/she cannot differentiate visual resolutions (e.g., replacing 5 with 5.1 on a visual plot may make no difference to users)" and "for interactive visualization there is an acceptable maximal latency for users".

We should continue such work, and continue to borrow and use HDI results from the HCI and visualization communities. At the same time, *many HILDA contexts that do not involve visualization seem to raise interesting and important HDI issues too, for which we have not paid sufficient attention.* For example, recent work on entity matching [10] observes that at the start of solving a problem (e.g., EM), users often do not even know the problem definition (e.g., what it means to be a match). More generally, user understanding is evolving during the data management process. If this is the case, then it can change the way we approach entity matching (e.g., we would now start by trying to help the user understand what it means to be a match).

As another example, recent work [6] has also observed that it is often easier for a user to recognize if a given query answers his/her need, than to come up with the query in the first place. More generally, it is easier for a user to recognize something than to find it. This observation was exploited to develop a solution to allow lay users to pose structured queries to a database [6].

As yet another example of HDI issues outside the context of visualization, today we do not really know how humans make decisions under "data uncertainty". Back in the 1940s there was a lot of interest in how humans make decisions under (probabilistic) uncertainty. For example, how should they choose between two choices: receiving $50 outright or with 0.5 probability receiving $200 and 0.5 probability losing $100? Expected utility theory was developed to address such problems, and this theory assumes humans behave rationally. However, in the 1970s and later, the work of Daniel Kahneman, Amos Tversky, and Richard Thaler [16, 25] shows that humans are not rational. They exhibit certain biases and behaviors when making decisions under uncertainty.

Thus, it would be very interesting to examine whether humans exhibit certain biases and behaviors when making decisions given incomplete or ambiguous data. For example, how would a crowd worker decide on matching two tuples if the data is incomplete or ambiguous? Would the wording of the instruction affect his/her decision? If so, how? Clearly, understanding these issues can help us design much better HILDA solutions.

As a result, I argue that we are still far away from a comprehensive HDI theory, that we should also pay more attention to HDI issues outside the HCI and visualization contexts, and that in the long run we should strive to codify all these HDI observations into a coherent and comprehensive "theory of human data interaction".

4 WHERE TO GO FROM HERE?

So far I have discussed the problems that are likely to fall under the scope of HILDA, and important aspects in developing HILDA solutions that I think our community should devote more attention to. I now take a step back and speculate about what may happen in the future.

First, it is clear from the discussion in this paper that the role of humans have steadily increased for the data management community. If much earlier, when working on RDBMSs, the role of humans was somewhat limited, then that situation has changed dramatically in the past ten years, with numerous works on human topics such as crowdsourcing, making data management easier for lay users, and with workshops such as the HILDA series. Looking forward, efforts to develop data and software-centric communities will only accelerate, and humans play even more important roles in building such communities. Thus, it appears that going forward data management can only involve more humans in more ways, not less, and that the HILDA problems will be an increasingly important part of the data management landscape.

In the short term, I believe the HILDA community is likely to seek more engagement with the AI, machine learning, HCI, and visualization communities. This is already happening, and is important. But I believe this will be far from enough. In a sense, this will give us more powerful solution techniques, as we learn from these communities. But as discussed earlier, it is critical that we focus on problems that real users care about and that we build systems and tools that real users are willing to use. Toward this goal, it is important that we also seek engagement with systems/tool communities, such as the vibrant and growing PyData/R communities, and build on their results where appropriate.

I mentioned the phrase "Human Data Interaction (HDI)" earlier. This phrase has actually been used to refer to what HILDA researchers are currently doing, but also to the management of personal data within a large data-centric ecosystem of users (who generate personal data) and companies (which want to process and use such data) [7], among others. In the longer term, there is a reasonable chance that all these efforts will coalesce to form a larger field called HDI, which sits at the intersection of many fields, including data management, data science, HCI, visualization, AI, psychology, and more.

This alludes to a danger that the HILDA community may face. It looks likely that many other communities also work on "human-in-the-loop" data management. So what distinguishes us? What do we bring to the table? How do we stay relevant? The answer, I believe, lies in the hope that we will not focus on just a particular technique or just a particular set of "point" problems. Rather, we will seek to solve "end to end" problems for real users, and build practical "end to end" systems. If we neglect these, we risk not having a clear identity and becoming increasingly irrelevant.

5 CONCLUSIONS

In this paper I have offered a personal perspective on some "big picture" aspects of HILDA. I argued that the role of humans can only grow for data management, and thus HILDA will make up an increasingly important part of the data management community.

I discussed a range of problems that I believe should fall under HILDA's scope, including some that have received relatively little attention, such as fostering data and software-centric communities. I discussed important aspects in developing HILDA solutions that I believe our community should pay more attention to. These include solving problems that real users care about, developing how-to guides to users, building end-to-end systems (such as building on top of the "Pandas system"), developing benchmarks and challenges, and developing a theory of human data interaction.

Finally, I speculated about the future of the field, and discussed the dangers it can face, given that many other communities are also working on related problems. I argued that a focus on end-to-end problems and systems is important for us to thrive and stay relevant.

Acknowledgment: I thank the reviewers for invaluable comments. This work is generously supported by WalmartLabs, Google, Johnson Control, American Family Insurance, UW-Madison UW2020 grant, NIH BD2K grant U54 AI117924, and NSF Medium grant IIS-1564282.

REFERENCES

[1] Daniel J. Abadi et al. 2014. The Beckman Report on Database Research. *SIGMOD Record* 43, 3 (2014), 61–70. https://doi.org/10.1145/2694428.2694441
[2] Christopher Ahlberg et al. 2003. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *The Craft of Information Visualization*. 7–13.
[3] Leilani Battle et al. 2017. Position Statement: The Case for A Visualization Performance Benchmark. In *DSIA Workshop*.
[4] Anant P. Bhardwaj et al. 2015. DataHub: Collaborative Data Science & Dataset Version Management at Scale. In *CIDR*.
[5] Matthew Brehmer and Tamara Munzner. 2013. A Multi-Level Typology of Abstract Visualization Tasks. *IEEE Trans. Vis. Comput. Graph.* 19, 12 (2013), 2376–2385.
[6] Eric Chu et al. 2009. Combining keyword search and forms for ad hoc querying of databases. In *SIGMOD*.
[7] Andy Crabtree and Richard Mortier. 2015. Human Data Interaction: Historical Lessons from Social Studies and CSCW. In *ECSCW The 14th European Conference on Computer Supported Cooperative Work*.
[8] Sanjib Das et al. 2017. Falcon: Scaling Up Hands-Off Crowdsourced Entity Matching to Build Cloud Services. In *SIGMOD*.
[9] AnHai Doan et al. 2007. User-Centric Research Challenges in Community Information Management Systems. *IEEE Data Eng. Bull.* 30, 2 (2007), 32–40.
[10] AnHai Doan et al. 2017. Human-in-the-Loop Challenges for Entity Matching: A Midterm Report. In *The 2nd Workshop on Human-In-the-Loop Data Analytics, HILDA@SIGMOD 2017*.
[11] AnHai Doan, Alon Y. Halevy, and Zachary G. Ives. 2012. *Principles of Data Integration*. Morgan Kaufmann.
[12] Philipp Eichmann et al. 2016. Towards a Benchmark for Interactive Data Exploration. *IEEE Data Eng. Bull.* 39, 4 (2016), 50–61.
[13] Yash Govind et al. 2017. CloudMatcher: A Cloud/Crowd Service for Entity Matching. In *BIGDAS*.
[14] Zachary G. Ives et al. 2015. Looking at Everything in Context. In *CIDR*.
[15] Jianfeng Jia et al. 2016. Towards interactive analytics and visualization on one billion tweets. In *GIS*.
[16] Daniel Kahneman and Amos Tversky. 1979. Prospect Theory: An Analysis of Decision Under Risk. *Econometrica* 47, 2 (1979), 263–292.
[17] Sean Kandel et al. 2011. Wrangler: interactive visual specification of data transformation scripts. In *CHI*.
[18] Sean Kandel et al. 2012. Enterprise Data Analysis and Visualization: An Interview Study. *IEEE Trans. Vis. Comput. Graph.* 18, 12 (2012), 2917–2926.
[19] Albert Kim et al. 2015. Rapid Sampling for Visualizations with Ordering Guarantees. *PVLDB* 8, 5 (2015), 521–532.
[20] Pradap Konda et al. 2016. Magellan: Toward Building Entity Matching Management Systems. *PVLDB* 9, 12 (2016), 1197–1208.
[21] Yongjoo Park et al. 2016. Visualization-aware sampling for very large databases. In *32nd IEEE International Conference on Data Engineering, ICDE 2016, Helsinki, Finland, May 16-20, 2016*.
[22] Peter Pirolli and Stuart K. Card. 1995. Information Foraging in Information Access Environments. In *CHI*.
[23] Ben Shneiderman. 2003. The eyes have it: a task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*. 364–371.
[24] Wang-Chiew Tan et al. 2018. Big Gorilla: an Open-source Ecosystem for Data Preparation and Integration. In *IEEE Data Engineering Bulletin, Special Issue on Data Integration*.
[25] Richard Thaler. 1980. Toward a Positive Theory of Consumer Choice. *Journal of Economic Behavior and Organization* 1, 1 (1980), 39–60.
[26] Eugene Wu et al. 2015. Towards perception-aware interactive data visualization systems. In *DSIA Workshop*.