Entity Resolution with Crowd Errors

Vasilis Verroios, Hector Garcia-Molina

Stanford University

{verroios, hector}@stanford.edu

Abstract—Given a set of records, an ER algorithm finds records that refer to the same real-world entity. Humans can often determine if two records refer to the same entity, and hence we study the problem of selecting questions to ask error-prone humans. We give a Maximum Likelihood formulation for the problem of finding the “most beneficial” questions to ask next. Our theoretical results lead to a lightweight and practical algorithm, bDENSE, for selecting questions to ask humans. Our experimental results show that bDENSE can more quickly reach an accurate outcome, compared to two approaches proposed recently. Moreover, through our experimental evaluation, we identify the strengths and weaknesses of all three approaches.

I. INTRODUCTION

Entity resolution (ER) detects records that represent the same real-world entity. Computer algorithms are often used for ER, but in many cases where records contain images or natural language, humans can more accurately tell if two records represent the same real-world entity. Google, Bing, and Facebook use this approach to create summary records for entities in web search or resolve duplicates for entities on the map (e.g., Facebook Places).

Still, humans can make mistakes. Even for humans, questions for some specific pairs of records are inherently difficult to answer correctly. Moreover, in large-scale crowdsourcing platforms, a non-negligible portion of workers are spammers, or provide low quality answers because they do not pay full attention to the tasks.

In this paper, we study the problem of entity resolution where evidence is collected on-demand from possibly erroneous humans. That is, given a set of records, we find which questions between pairs of records will reveal “the most” about the underlying entities. Our solution also takes into account evidence from traditional record similarity functions. (For instance, a computer can compare the author, title and venue fields of records representing two articles to determine with some confidence if the records represent the same article.)

A brute force approach would be to consider every possible pair of records in the data set, and to ask a human if that pair represents the same real world entity. Since humans can make mistakes, we would further repeat the same question to multiple humans until we were confident we had the correct answer for that question. While this scheme would produce the most evidence, it is clearly infeasible when more than a handful of records are involved: Humans take time to answer questions and usually have to be paid. Instead, our approach (Fig. 1) is to first generate as much computer evidence as possible, which we assume is a fast and inexpensive process (if not very accurate). Then, based on the evidence on hand, we select one or more pairs of records, and ask humans for their answers. After incorporating the new evidence, we repeat the process, ending when we are “confident enough” that we can partition the records into entities; or when we exhausted our budget. At that point, we apply an entity resolution algorithm (ERA): the ERA takes the probabilistic evidence and partitions the records, where each cluster represents the records thought to represent one real world entity. We base our Question Selection (QS) on a heuristic driven by a number of interesting theoretical results that capture the usefulness of each question. The details are presented in the rest of this paper.

The main difference of our approach from others is the way we handle human errors. Some papers (like [1], [2], [3]) assume that humans make no mistakes. In such a scenario, the only uncertainty comes from the computer generated similarities, so it becomes easier to reason about what questions to ask. Most techniques [4], [5], [6] for dealing with human errors, assume that humans make no mistakes. In such a scenario, the only uncertainty comes from the computer generated similarities, so it becomes easier to reason about what questions to ask. Most techniques [4], [5], [6] for dealing with human errors, end up in repeating a question to n humans, and making n large enough so that the “majority” answer is effectively error-free. As we will see in our experiments, this approach works in some cases, but in others may be wasteful. Another paper [7] does take into account human errors, but selects record pairs to resolve next at random.

Our solution, on the other hand, determines questions to ask based on the evidence collected (both machine and human evidence), in order to increase the probability of the Maximum Likelihood(ML) clustering. In a sense, our approach finds the best possible questions to ask, independent of the ERA. Our scheme is not constrained to ask fixed blocks of n questions for each pair of records, and is free to allocate questions as it best sees fit.

In addition to our question selection strategy, in this paper we also present an interesting ERA algorithm that works especially well with probabilistic evidence. We call this algorithm SCC (Spectral-Connected Components). A traditional ERA would first apply a probability threshold to convert the evidence into a deterministic graph. For example, nodes x and y are connected in the graph if the probability that they are the same entity is higher than say 0.9. Then the traditional ERA would apply some global clustering algorithm to partition the...
graph. On the other hand, SCC forms clusters by examining the full evidence. For instance, to decide if two clusters should merge into one, it considers all questions that involve records in the two clusters in question. As we will see, SCC yields high quality results, not just when used with our own question selection strategy, but also when used with other selection strategies.

In summary, we make the following contributions:

- We provide a formal definition for the question selection problem, under the assumption that human answers may contain errors, in Section III-C.
- We provide theoretical results that identify “beneficial” questions with a reasonable computational cost, in Section III.
- We propose a lightweight algorithm that applies our theoretical results to select questions, in Section IV.
- We present our SCC clustering algorithm for partitioning records based on probabilistic evidence, in Section V.
- We compare our question selection algorithm to two algorithms proposed in related work, using datasets of images and AMT workers, and we identify in which cases each algorithm is better, in Section VI.

A. Running Example

![Fig. 2. Entity resolution in a set of four images](image)

Consider the dataset of 4 records in Fig. 2. The same person appears in images a and b and another person appears in images c and d. We are provided with 5 answers: a) YES from a human for the pair (a, b), b) YES from a human for the pair (c, d), c) YES from an image processing algorithm for the pair (c, d), d) NO from an image processing algorithm for the pair (a, d), and e) NO from an image processing algorithm for the pair (b, c). In this example, say humans make mistakes 80% of the time. Therefore, along with each human answer we have a probability of being correct, denoted by $p^H$, equal to 0.8. In addition, say the image processing algorithm generates its three answers with confidence 0.6. Thus, we associate this value, denoted by $p^M$, with its answers. Our objective is to select which additional questions to send to humans.

II. PRELIMINARIES

A. Model

Our model assumes that between pairs of records we have YES/NO answers, from machines or humans. A YES answer is correct, if the two records actually refer to the same entity, while a NO answer is correct, if the two records actually refer to different entities. Note the subtle difference between this model and the model of [3]. To illustrate, consider three records $a$, $b$, and $c$ that refer to a single entity. In [3], a human, who is always correct, can answer NO to the pair $a-b$, as long as he or another human answers YES to $b-c$ and to $c-a$ (so that the correct entity can be discovered through transitive closure). In our case, the correct answer to all three questions is YES because the records are part of the same entity, independent of how entities are discovered by the ERA.

Along with each answer, there is a probability of the answer being correct. For human answers, we assume that there is a single probability for any record-pair answer provided by a human. We call this probability $p^H$. In a specific crowdsourcing platform, $p^H$ can be the expected accuracy of humans for a specific type of dataset.

In practice, some record pairs may be harder than others for humans to answer. However, we make the single $p^H$ assumption to simplify the question selection reasoning. In our experimental evaluation we show that in data sets where the assumption does not hold, our approach still quickly converges to high quality results.

We assume that humans involved in the process are not malicious, i.e., the worst a human can do is guess at random. Thus, $p^H$ is never lower than 0.5. For machine answers, there can be a different probability for each record-pair answer. Since most machine algorithms generate a similarity value between two records, this similarity value needs to be translated to a probability, $P$, of the two records referring to the same entity. If $P$ is lower than 0.5, then we need to generate a NO answer with probability of being correct $1 - P$; otherwise, we will generate a YES answer with probability $P$.

B. Maximum Likelihood Clustering

In this section, we define the Maximum Likelihood (ML) clustering, which will be a main building block in the definition of the “best” next question, in the next section.

Let us denote the answers which already appear in our dataset as evidence $E$. In Fig. 2, $E$ consists of five answers. First, we will examine the most probable clustering of records in Fig. 2 based on $E$, such that all records of each formed cluster refer to one entity. An example of such a clustering appears in Fig. 3(a) where all the records are clustered into a single entity.

The probability of each clustering $C_k$ given the evidence $E$, is $P(C_k|E)$, and we want to find

$$\max_k P(C_k|E) = \max_k \frac{P(C_k, E)}{P(E)} = \max_k \frac{P(E|C_k)P(C_k)}{P(E)}$$  \(1\)

Therefore, we want to find the clustering $C_k$ that maximizes $P(E|C_k)P(C_k)$.

We assume that each clustering has the same a priori probability, $P(C_k)$. In other words, we assume that we don’t have any other prior knowledge about the dataset, except for the evidence $E$. Other prior knowledge that could have been available, is statistics regarding our dataset, e.g., distribution
for the number of records referring to the same entity. In our technical report [9] we discuss some possible extensions to our model, for future work.

Since \( P(C_k) \) is the same for any clustering \( C_k \), we will simply have to find the clustering maximizing \( P(E|C_k) \), i.e., the ML clustering. For example, consider the clustering \( C_1 = \{ \{a, b, c, d\} \} \) depicted in Fig. 3(a) \( P(E|C_1) = 0.8^2 \cdot 0.6^2 \cdot (1 - 0.6)^2 \), i.e., answers about the pairs, \( (a, b), (c, d), (a, d) \) and \( (b, c) \) are proven to be correct, while the answer for \( (b, d) \) is proven to be false, if \( C_2 \) is the “true” clustering.

\[ \text{ML clustering} \]

A clustering with a higher likelihood \( P(E|C_k) \) is clustering \( C_2 = \{ \{a, b\}, \{c, d\}\} \), depicted in Fig. 3(b) \( P(E|C_2) = 0.8^2 \cdot 0.6^2 \cdot (1 - 0.6)^2 \), i.e., answers about the pairs, \( (a, b), (c, d) \) and \( (a, d) \) and \( (b, c) \) are proven to be correct, while the answer for \( (b, d) \) is proven to be false, if \( C_2 \) is the “true” clustering. Note that \( C_2 \) is the ML clustering for the five answers we are given, i.e., no other clustering can do better.

For the rest of the paper, we will denote the probability \( P(E|C_{ML}) \) of the ML clustering, \( C_{ML} \), by \( \mathcal{ML}(E) \), i.e.,

\[ \mathcal{ML}(E) = \max_k P(E|C_k) \]

Finding the ML clustering is NP-hard, by reduction from the ML clustering. For example, consider the clustering \( C_k \) depicted in Fig. 3(b). For instance, in Fig. 3(b), the YES between \( a \) and \( b \), denoted by \( n \), is proven to be false, if \( C_2 \) is the “true” clustering. Note that \( C_2 \) is the ML clustering for the five answers we are given, i.e., no other clustering can do better.

C. Next Single Question

In this section, we give a definition for the “best” single question to ask next. Using this definition, we unfold a number of theoretical results, in Section III, that form the basis of our algorithms, described in Section IV.

Let us examine which is the “best” question to ask next. For a question \( q_{ij} \) between records \( i \) and \( j \), we may get back a NO, denoted by \( n_{ij} \), or a YES, denoted by \( y_{ij} \). We define the Expected Probability (EP) of the ML clustering (technically we start from the Maximum A Posteriori(MAP) clustering), over the cases of a YES and a NO, for a question \( q_{ij} \) as:

\[ EP_{ij} = P(y_{ij}|E) \max_k P(C_k|E \land y_{ij}) + P(n_{ij}|E) \max_k P(C_k|E \land n_{ij}) \]

In case of a YES, we have:

\[ P(y_{ij}|E) \max_k P(C_k|E \land y_{ij}) = \frac{P(E \land y_{ij}) \max_k P(C_k|E \land y_{ij})}{P(E)} \]

Likewise, in case of a NO we have:

\[ P(n_{ij}|E) \max_k P(C_k|E \land n_{ij}) = \frac{1}{P(E)} \max_k P(E \land n_{ij}) \]

Since, we assume equal a priori clustering probabilities, i.e., \( \forall k, P(C_k) = P_{C} \), we have:

\[ EP_{ij} = \frac{P_{C}}{P(E)} \left( \max_k P(E \land y_{ij}|C_k) + \max_k P(E \land n_{ij}|C_k) \right) \]

\[ = \frac{P_{C}}{P(E)} \left( \mathcal{ML}(E \land y_{ij}) + \mathcal{ML}(E \land n_{ij}) \right) \]

Therefore, the “best” single question to ask next, is the question \( q_{ij} \) that maximizes:

\[ \alpha(ij) = \mathcal{ML}(E \land y_{ij}) + \mathcal{ML}(E \land n_{ij}) \]

\( \alpha(ij) \) is the probability, in expectation, of the MAP clustering for the new evidence (including the answer for \( q_{ij} \)), divided by the constant-for-all-questions quantity of \( \frac{P_{C}}{P(E)} \).

III. Theoretical Results

In this section, we discuss our main theoretical results: THEOREM 1 and its three COROLLARIES. The results from the corollaries are directly applied by our algorithm in Section IV. We first provide an overview before discussing the details. Each corollary applies to a particular scenario, and provides suggestions for good questions to ask in each case. COROLLARY 1 applies to the scenario of Fig. 4(a) where we can identify three sets of records, \( S_i \), \( S_j \), and \( R = \{S_i \cup S_j\} \), and there are no answers between pairs from different sets. \( R \) is the set of all records.) COROLLARY 1 states that in this case, any question between two records from different sets is one of “best” questions to ask. COROLLARY 2 applies to a more general case, depicted in Fig. 4(b) where there are NO answers between records of \( S_i \) or \( S_j \) and \( R = \{S_i \cup S_j\} \). In this scenario, a question between an \( S_i \) record and an \( S_j \) record is one of the “best” questions to ask. Extending even further the scenario of COROLLARY 2 we add a) YES answers between \( S_i \) or \( S_j \) and \( R = \{S_i \cup S_j\} \) and b) YES/NO answers between \( S_i \) and \( S_j \), in Fig. 4(c) In this case, COROLLARY 3 provides guarantees on how “good” or “bad” a question between \( S_i \) and \( S_j \) can be, based on the “weights” of the added answers.

We start with some preliminary results and definitions that will simplify the statements and proofs of THEOREM 1 and COROLLARIES 1, 2, and 3. While in our definition of the “best” question to ask next, the NP-hard problem of ML
clustering is involved, our main results, in COROLLARIES 1, 2, and 3, do not assume knowledge of the ML clustering. The proofs for all the lemmas, theorems and corollaries in this section can be found in our technical report [9].

**Lemma 1.** \( \forall q_{ij}, \alpha(ij) \in [\mathcal{ML}(E), 2p^H \mathcal{ML}(E)] \)

Since human participants are not malicious, \( p^H \geq 0.5 \), it follows that \( 2p^H \mathcal{ML}(E) \geq \mathcal{ML}(E) \). Intuitively, a question on the lower bound, i.e., with \( \alpha = \mathcal{ML}(E) \), does not increase our confidence for the ML clustering. On the other hand, any question with \( \alpha > \mathcal{ML}(E) \), i.e., strictly higher than the lower bound, will increase, even slightly, our confidence in the ML clustering; whether the question is answered YES or NO. We will call a question with an \( \alpha \) strictly higher than the lower bound, **beneficial**. Now, the most beneficial questions that achieve the upper bound, \( 2p^H \mathcal{ML}(E) \), are called **optimal**, in the sense that no other question can do better.

**Definition 1.** **Beneficial question:** A question with an \( \alpha \) strictly higher than \( \mathcal{ML}(E) \), is a beneficial question.

**Definition 2.** **Optimal question:** A question with an \( \alpha \) equal to \( 2p^H \mathcal{ML}(E) \), is an optimal question.

For instance, \( q_{ac} \) is a beneficial question in our running example of Fig. 2. A NO gives \( C_2 \) as the ML clustering, as shown in Fig. 5(a), with

\[
\mathcal{ML}(E \cap n_{ac}) = p^H_c P(E|C_2) = p^H_c \mathcal{ML}(E)
\]

(a) \( C_2 \) after a NO for \( q_{ac} \)

Fig. 5. Clustering pairs \( C_2 \) and \( C_1 \) after \( q_{ac} \) is answered

A YES gives \( C_1 \) as the ML clustering, as shown in Fig. 5(b), with

\[
\mathcal{ML}(E \cap y_{ac}) = p^H_c P(E|C_1) = p^H_c \frac{0.6 \times (1 - 0.6)^2}{0.6^2 \times (1 - 0.6)} \mathcal{ML}(E)
\]

\[
\mathcal{ML}(E|C_1) = 0.82 \times 0.6 \times (1 - 0.6)^2 \quad \text{and} \quad \mathcal{ML}(E) = 0.8 \times 0.6^2 \times (1 - 0.6)
\]

Hence, the probability of a correct answer from a machine, \( p^H_c \), for the two machine answers, \( n_{ac} \), is

\[
\alpha(ac) = p^H_c \frac{1 + 0.6 \times (1 - 0.6)^2}{0.6^2 \times (1 - 0.6)} \mathcal{ML}(E) \approx 1.33 \mathcal{ML}(E)
\]

On the other hand, a question which is not beneficial is question \( q_{ab} \), because

\[
\alpha(ab) = \mathcal{ML}(E \cap y_{ij}) + \mathcal{ML}(E \cap n_{ij}) = p^H_c \mathcal{ML}(E) + (1 - p^H_c) \mathcal{ML}(E) = \mathcal{ML}(E)
\]

In this example, the pair of records \((a, b)\) seems to refer to one entity and the pair \((c, d)\) also seems to refer to one entity. Still, it is not clear if \((a, b)\) refers to a different entity than \((c, d)\), or if all four records refer to one single entity. Thus, asking \( q_{ac} \) is more “useful” than asking \( q_{ab} \). This notion of “uncertainty” is formally defined by **balance**.

**Balance** expresses how “uncertain” we are about two clusters of an ML clustering. Should they be merged into one single entity or should they remain two separate clusters/entities? The higher the value of \( \lambda \), the higher the uncertainty. When \( \lambda \) reaches its maximum value, 1.0, we practically have no evidence for keeping the two clusters as two separate entities. On the contrary, a \( \lambda \) value of 0.0 indicates that the two clusters should definitely be kept as two separate clusters/entities. Below, we formally define \( \lambda \)-**balance** and we prove that \( \lambda \in [0.0, 1.0] \).

**Definition 3.** **\( \lambda \)-balance:** Consider an ML clustering with two clusters \( c_1 \) and \( c_2 \), as given in Fig. 6(a). Between \( c_1 \) and \( c_2 \), we have a set of YES answers, \( Y \), and a set of NO answers, \( N \), from humans and machines; for each answer of \( Y \) and \( N \) one record is in \( c_1 \) and the other in \( c_2 \). An answer \( a \in \{Y \cup N\} \) has a probability of being correct \( p_c(a) \). We denote:

\[
P_Y = \prod_{a \in Y} p_c(a), P'_Y = \prod_{a \in Y} (1 - p_c(a)) \quad (8)
\]

\[
P_N = \prod_{a \in N} p_c(a), P'_N = \prod_{a \in N} (1 - p_c(a)) \quad (9)
\]

and we say that there is a \( \lambda \)-**balance** between \( c_1 \) and \( c_2 \), for \( \lambda = \frac{P'_N}{P'_N + P'_Y} \).

**Lemma 2.** In a \( \lambda \)-**balance** between two clusters of an ML clustering, \( \lambda \in [0.0, 1.0] \).

In the ML clustering of our running example, in Fig. 3(b), between the two clusters \{a, b\}, \{c, d\}, we have \( N = \{n_{ad}, n_{bc}\}, Y = \{y_{bd}\} \), \( P_N = 0.6^2, P'_N = (1 - 0.6)^2, P_Y = 0.6 \) and \( P'_Y = (1 - 0.6) \). A \( \lambda \)-**balance** exists between the two clusters, for \( \lambda = \frac{(1-0.6)^2 + 0.6}{0.6^2 + (1-0.6)^2} = \frac{3}{4} \). In case the probability of a correct answer from a machine, \( p^M_c \), for the three machine answers, \( n_{ad}, n_{bc}, \) and \( y_{bd}, \) was lower than 0.6, then the \( \lambda \)-**balance** between \{a, b\} and \{c, d\}, would be higher than \( \frac{3}{4} \).

On the other hand, if \( p^M_c \) was higher than 0.6, \( \lambda \) would be lower than \( \frac{3}{4} \). In Fig. 6(b) we plot the \( \lambda \)-**balance** between \{a, b\} and \{c, d\}, as \( p^M_c \) goes from 0.5 to 1.0.

The following theorem connects the notions of \( \lambda \)-**balance** and the \( \alpha \) value of a new question. While the theorem assumes knowledge of the ML clustering, we present three corollaries that do not assume such knowledge. These corollaries identify practical cases for finding beneficial questions, since finding the ML clustering is NP-hard. In Section IV we will present an algorithm that makes use of the results from the corollaries.
THEOREM 1. If an ML clustering contains two clusters with a \( \lambda \)-balance between them, for \( \lambda > \frac{1-p_c^0}{p_c^1} \), any question between a record \( i \) from the first cluster and a record \( j \) from the second cluster, is a beneficial question. In addition, 
\[
\alpha(ij) \geq (1 + \lambda)p_c^1 \text{ML}(E)
\]

An application of THEOREM 1 appears in the running example, in Fig. 4(b) where there is a \( \lambda \)-balance between the two clusters, \( \{a, b\}, \{c, d\} \), of the ML clustering, for \( \lambda = (1-0.6)^2+0.6 \). Questions \( q_{ac}, q_{ad}, q_{bc}, q_{bd} \) are beneficial with an \( \alpha \) of, at least, \( (1+\lambda)0.8 \text{ML}(E) \approx 1.33 \text{ML}(E) \); as we discussed before, \( \alpha \) is exactly that for these 4 questions.

The three following corollaries are based on the notion of \( \lambda \)-balance, and define cases where we can “easily” find optimal or beneficial questions. During the statements and proofs of the three corollaries, we use the following notation:

- \( \mathcal{G} \): The graph formed by the YES and NO answers from machines and humans collected so far. The nodes of the graph are the records of the dataset and an edge between two records, consists of an answer from a human or machine for the two records.
- \( \mathcal{G}_Y \): \( \mathcal{G} \) without the NO edges.

COROLLARY 1. If \( \mathcal{G} \) has two or more connected components, a question between any record \( i \) from one connected component and any record \( j \) from another connected component, is an optimal question.

Fig. 4(a) depicts the case where COROLLARY 1 applies. The proof (found in our technical report [9]) states that there will always be an ML clustering having records \( i \) and \( j \) in two different clusters, with a 1.0-\( \lambda \)-balance between them. Note that each connected component is not necessarily a cluster in an ML clustering, i.e., there may be two or more clusters of an ML clustering inside each connected component.

COROLLARY 1 can be generalized to the case of COROLLARY 2, illustrated in Fig. 4(b).

COROLLARY 2. Consider two connected components of \( \mathcal{G}_Y \) without any NO answers between a record from the first and a record from the second component. If such pair of components exist, a question between any record of the first component and any record from the second one, is an optimal question.

In order to find an optimal question, COROLLARY 2 suggests that we can 1) remove all the NO answers from \( \mathcal{G} \), 2) find the connected components, and 3) try all the pairs of connected components until we find two components without any NO answers between a record of the first component and a record of the second one. If we find such a pair of components, we can arbitrarily select one record from the first component and one record from the second component and issue a question for this record pair. In Section IV we discuss in detail the complete algorithm that uses the corollaries presented here.

Now, let us consider a more general case, shown in Fig. 4(b)

except for \( a \) a YES answer with probability of being correct \( p_c = 0.6 \), between the \( S_i \) and the “Rest of the Graph” and \( b \) a NO answer between \( S_i \) and \( S_j \), again with \( p_c = 0.6 \). How “bad” can a question between arbitrarily picked records \( i \) from \( S_i \) and \( j \) from \( S_j \) be, in this case? Or, in other words, is there a lower bound for \( \alpha(ij) \)? Such a guarantee would help us find beneficial questions. As COROLLARY 3 states, \( q_{ij} \) is a beneficial question, with \( \alpha(ij) \geq (1 + \frac{1-0.6^2}{0.6})p_c^1 \text{ML}(E) \). Next, we give the general definition for the ratio \( \frac{1-0.6^2}{0.6} \) resulting from answers \( a \) and \( b \), in this example.

Fig. 7. COROLLARY 3 description
DEFINITION 4. \( \rho \)-ratio: Consider two disjoint subsets of records, \( S_i \) and \( S_j \), of the overall set of records \( R \), as they appear in Fig. 7(b) We denote a) the YES answers between \( S_i \) and \( R - \{S_i \cup S_j\} \) by \( Y_1 \), b) the YES answers between \( S_i \) and \( R - \{S_i \cup S_j\} \) as \( Y_2 \), c) the NO answers between \( S_i \) and \( S_j \) as \( N \), and d) the YES answers between \( S_i \) and \( S_j \) as \( Y \). The probability of an answer \( a \in \{Y_1 \cup Y_2 \cup N \cup Y\} \) being correct, is \( p_c(a) \). In addition, we denote 1) \( P_{Y_1} = \prod_{a \in Y_1} p_c(a) \), 2) \( P_{Y_2} = \prod_{a \in Y_2} p_c(a) \), 3) \( P_N = \prod_{a \in N} p_c(a) \), 4) \( P_Y = \prod_{a \in Y} p_c(a) \), 5) \( P_{Y_1}' = \prod_{a \in Y_1} (1 - p_c(a)) \), 6) \( P_{Y_2}' = \prod_{a \in Y_2} (1 - p_c(a)) \), 7) \( P_N' = \prod_{a \in N} (1 - p_c(a)) \), and 8) \( P_Y' = \prod_{a \in Y} (1 - p_c(a)) \).
We call \( \rho \)-ratio, the ratio:
\[
\frac{P_{Y_1}P_{Y_2}'}{P_{Y_1} + P_{Y_2}} = \min\left\{\frac{P_N'}{P_{Y_1}}, \frac{P_Y'}{P_Y}\right\}
\]
COROLLARY 3. When between two disjoint subsets of records, \( S_i \) and \( S_j \), there is a \( \rho \)-ratio with
\[
\rho > \frac{(1 - p_c^1)}{p_c^0}
\]
a question between an arbitrary record \( i \) of \( S_i \) and an arbitrary record \( j \) of \( S_j \) is a beneficial question.
Moreover,
\[
\alpha(ij) \geq (1 + \rho)p_c^1 \text{ML}(E)
\]

IV. QUESTION SELECTION ALGORITHM
As COROLLARY 3 points out, we can find a beneficial question, by detecting two disjoint sets of records that a) are not “strongly” connected with YES answers with the rest of the dataset and b) between the two sets there is neither “strong” YES evidence, nor “strong” NO evidence. The notion of “strong” is quantified by the \( \rho \)-ratio. In other words, COROLLARY 3 suggests a way to detect “lack”-of-evidence inside the dataset.
the formal sense), it detects “lack”-of-evidence and picks questions that strongly enhance the pre-existing evidence. In the experimental evaluation, we show that as we keep asking questions that our heuristic selects, we rapidly converge to high precision and recall, even when a simple ERA is applied instead of the ML clustering.

Our heuristic is a lightweight algorithm based on the $\rho$–ratio. The algorithm’s main goal is finding the two sets of records with the highest $\rho$–ratio. Although the algorithm directly applies corollary in two aspects:

- **The algorithm may not find the two sets of records with the highest $\rho$–ratio:** One possibility would be to examine all possible pairs of disjoint sets of records and find the pair with the maximum $\rho$–ratio. However, the computational overhead of this approach makes it infeasible, since the number of such pairs is exponential to the number of records in the dataset. Therefore, our heuristic only examines some of all the possible pairs of record sets.

- **The $\rho$–ratio found, may be less than the $\frac{(1-p^*)}{p^*}$ threshold:** Even if the algorithm finds the pair of record sets with the highest $\rho$–ratio, this ratio may be less than the threshold corollary states. In the case the $\rho$–ratio found is less than the threshold, a question between the two record sets will not formally be a beneficial question. Still, such question lies on a part of the dataset where “lack”-of-evidence is detected, as the high $\rho$–ratio points out.

### Algorithm 1 DENSE

**Input:** $G$: The graph of answers  
**Output:** candidates: the pairs of record sets examined by the algorithm

1. Pick the pair of records with the highest $\rho$–ratio between them
2. Add this pair to candidates
3. while There are more than one sets of records do
4. Pick the answer with the highest YES probability between records in different sets
5. Merge the two sets of the two records into one
6. Compute the $\rho$–ratio between the merged set and every other set
7. Add to candidates, the merged set and the set that achieved the highest $\rho$–ratio
8. end while
9. Pick the pair of record sets with the highest $\rho$–ratio, from candidates
10. Between these two record sets, randomly pick one record from the first set and one from the second set
11. question := a question between these two records

Alg. 1 describes our heuristic. We start with each set of records consisting of a single record and we compute the $\rho$–ratio between every pair of records (lines 1–2). Then, we pick the pair of records having a YES answer with a probability closer to 1.0 than any other pair of records (line 4). We merge the two records into one set and, then, for every other record, we compute the $\rho$–ratio between the merged set and that record (lines 5–6). We continue to merge record sets and compute the $\rho$–ratios between the currently merged record set and the rest of the record sets, until all records are merged into a single set (lines 3–8). In the end, we select the pair of record sets with the highest $\rho$–ratio, which was found during the process (line 9). The question we select is between two arbitrarily chosen records, one from the first set and one from the second set (lines 10–11).

### Algorithm 2 bDENSE

**Input:** candidates: the pairs of record sets examined by the DENSE algorithm  
**Output:** questions: a set of questions to ask next
1. Sort the pairs of record sets in candidates, in descending order, based on the $\rho$–ratio between them
2. involvedRecs = $\emptyset$
3. while candidates is not empty do
4. Remove the pairs of record sets in the 1st position of candidates
5. if Any of the records in these two sets is already in involvedRecs then
6. continue to line 3
7. end if
8. Between these two record sets, find the pair of records that has an answer with a probability closest to 0.5
9. Add a question between these two records in questions
10. Add the records from the two record sets in involvedRecs
11. end while

In addition to Alg. 1 which selects a single question to ask next, we propose a batch version of our heuristic in Alg. 2. The reason for having a batch version is twofold. First, when we issue multiple questions, we allow many humans to answer questions in parallel. Second, Alg. 1 follows a greedy approach that attempts to find the single best question to ask next. If Alg. 1 finds a pair of record sets with a high $\rho$–ratio, it may keep asking questions between these two record sets, for a large number of consecutive invocations. However, a better strategy may be to explore other record sets of the dataset that may also have a high $\rho$–ratio, and not just focus on the pair with the highest $\rho$–ratio.

In the experimental results of Section VI, we will use bDENSE, the batch version of our heuristic.

### V. ENTITY RESOLUTION ALGORITHMS (ERA)

As discussed in the Introduction, the Entity Resolution Algorithm (ERA) is applied after questions have been posed to the crowd. Based on the evidence collected, the ERA returns a partition of the records, where each cluster represents what is believed to be an entity. One option is for the ERA to return the Maximum Likelihood clustering. We have argued that the ML clustering is the best guess one can make, so in a sense this solution is the best possible. Unfortunately, finding the ML clustering...
is prohibitively expensive except in the smallest scenarios. Thus, in practice an ML ERA is not used; instead heuristic approximations are used to find good clusterings.

It is outside the scope of this paper to survey all possible ERAs. Instead we discuss one simple representative algorithm, Transitive Closure (TC). While there are better algorithms (that would take longer to explain), TC represents a big class of algorithms that apply a probability threshold to obtain a deterministic graph of records, which is then partitioned.

We also present a new ERA that we believe is better suited to partitioning records with the type of evidence we have (without first generating a deterministic graph). While this ERA is a natural extension of TC, as far as we know it has not been discussed in the entity resolution literature.

Note incidentally that our question generation algorithm (Section IV) selects questions that improve the ML clustering (without actually computing the ML clustering). Of course, such questions may not be the best to improve a clustering obtained via a heuristic ERA. Nevertheless, in Section VI we show that our questions are indeed useful even when the heuristic algorithms are used.

A. Transitive Closure (TC)

The TC algorithm uses a probability threshold to determine which pairs of records refer to the same entity. After identifying such pairs, it applies the transitive relation to connect records into components (clusters). For example, say for a pair, \(a\) and \(b\), we have a YES with a probability of being correct, higher than the threshold, and the same applies to a pair, \(b\) and \(c\). Then we can infer that \(a\) and \(c\) also refer to the same entity, and hence all three records are in the same cluster.

An issue with TC (and other schemes that apply thresholds) is that it can be misled by erroneous answers. To illustrate, consider the scenario in Fig. 8(a). Entity \(E_1\) consists of 5 records and entity \(E_2\) also consists of 5 records. Therefore, there are 25 pairs of records with one record from \(E_1\) and one record from \(E_2\). Since humans make errors, there is a high likelihood one or more record pairs, out of the 25 pairs, will end up having one or more YES answers and zero NO answers. Just a single pair that happens to have a combined YES probability higher than the threshold will cause TC to incorrectly merge \(E_1\) and \(E_2\). As the number of constituent records increase, the chances that one record pair causes an erroneous merge increase.

Fig. 8. SCC description

B. Spectral-Connected-Components (SCC)

The SCC ERA avoids the problem of Fig. 8(a) by only merging clusters like \(E_1\) and \(E_2\) if the combined evidence of all 25 pairs indicates a high probability that the clusters represent the same entity.

SCC starts from the pair of records having the highest probability of being the same entity, given the answers for the two records. If this probability is higher than 0.5, SCC merges the two records into one component (cluster). In each step, SCC finds the pair of components with the highest probability of being the same entity, given the answers between them. If this probability is higher than 0.5 the two components are merged into one. Otherwise, SCC stops merging components, and returns as output the current set of components.

Fig. 8(b) gives an example of how SCC determines if it should merge two clusters or not. SCC examines the probability of \(E_1\) and \(E_2\) being a different entity (event \(dif\)), given the 4 answers between \(E_1\) and \(E_2\), denoted by \(A\). Here, \(P(dif|A)\) is:

\[
P(A|\text{dif}) = \frac{0.3 \cdot 0.4 \cdot 0.4 \cdot 0.9}{0.7 \cdot 0.6 \cdot 0.6 + 0.1 \cdot 0.3 \cdot 0.4 \cdot 0.9} 
\approx 0.63
\]

SSC assumes that \(P(\text{same}) = P(dif)\), i.e., answers involving records that do not belong to \(E_1\) or \(E_2\) are not taken into account for the merging decision. In each step, SCC finds the pair of clusters with the highest \(P(\text{same}|A)\). If \(P(\text{same}|A) > P(dif|A)\), SCC merges the two clusters into one. In this example, merging would not take place, since \(P(dif|A) > P(\text{same}|A)\).

VI. EXPERIMENTAL RESULTS

First, we give a brief overview of the alternative algorithms and the metric used, and then we discuss the experimental settings and our findings.

Question Selection Algorithms: We compare bDENSE (Alg. 2) with HALF, the heuristic proposed in [3], and MLF the Maximum-Likelihood-First strategy proposed in [2]. (Note that a heuristic similar to MLF is proposed in [11].)

HALF picks in each step the most “uncertain” pair of records, i.e., the pair with the-closest-to-0.5 probability of being the same entity. Note that the probability of being the same entity, is computed using only local information, i.e., the answers for a specific pair of records. For example, consider \(a\) a pair of records with a NO answer and a probability of the answer being correct 0.6 and \(b\) a pair of records with a YES answer and a probability of the answer being correct 0.7. HALF will first pick pair \(a\) to issue a question.

MLF sorts all pairs of records based on their probability of being the same entity. Again, just like HALF, this is the probability given only the answers for a specific pair of records, and not the global information for the dataset. The MLF strategy starts asking questions from the pair of records with the highest probability of being the same entity, and skips pairs that can be inferred to refer to the same or different entities. The inference is performed via the transitive relations on the human answers collected, up to the current step. For example, if for a pair of records, \(a\) and \(b\), we have a YES human answer for this pair, and for a pair, \(b\) and \(c\), we also have a YES human answer, then we can infer that \(a\) and \(c\) also refer to the same entity. In addition, if we get a NO answer for
pair $b-d$, we can infer that $a$ and $d$ refer to different entities, as well as that $c$ and $d$ refer to different entities.

Note that [2] (where MLF is defined) and [3] (where HALF is defined) assume that humans do not make errors. Since in our experiments humans do make mistakes, we extend MLF and HALF in the following way. When MLF or HALF select a pair of records to be evaluated by a human, we actually ask a sequence of humans to evaluate the same pair, until the pair is “fully resolved”. To illustrate, assume that for pair of records $(a, b)$, we have received two YES answers and one NO answer, i.e., our local evidence is $y_{ab} \land y_{ab} \land n_{ab}$. We then compute two probabilities: the probability that $a$ and $b$ are the same entity given this local evidence, and the probability that $a$ and $b$ are different entities again given this evidence. (For this computation we use the expected human accuracy, $p^H_{i,t}$.) If either of these probabilities is greater than some threshold, we say that $(a, b)$ is fully resolved and we stop asking questions for this pair. We use two different thresholds for pair resolution, 0.99 and 0.9999. The corresponding algorithms are HALF99, HALF9999, MLF99, and MLF9999.

**Entity Resolution Algorithms (ERA):** We use TC and SCC, the ERAs discussed in Section V in our experiments. In particular, TC is used with MLF and SCC is used in all other cases. The threshold TC uses is the same with MLF.

**Metric:** We use the F1 score to evaluate the quality of the evidence collected by each question selection algorithm. After a given number of answers collected by a question selection method, we apply the ERA algorithm. Then, we compare the output to the gold standard clustering. Specifically, we compare the pairs of records that refer to the same entity in the output, to the pairs referring to the same entity in the gold standard. We get the precision($p$) and recall($r$) by comparing the two sets of pairs and we compute the F1 score, $\frac{2pr}{p+r}$.

### A. Real Data

In this section, we present our experiments with two datasets of images, AllSports [11] and Gymnastics [12], and a third dataset containing CAPTCHAs [13].

1) Images Datasets: The AllSports [11] dataset contains 267 athlete images from 10 different sports, with each image showing a single athlete. In datasets like AllSports and Gymnastics, image processing algorithms are still far less accurate than humans. A typical approach for gathering evidence in such cases, is to use a low-cost crowdsourcing platform to gather some initial evidence for every image pair, and then use a smaller pool of trusted annotators to accurately evaluate the critical pairs. We emulate this approach by asking 10 inexperienced workers (no specific requirements) from Amazon Mechanical Turk (AMT [8]) to compare each two same-sport images, and generate an initial answer for that image pair. For example, if for a specific pair we get 6 NOs out of 10, we generate an initial NO answer with an associated probability of being correct of 6/10. (If all workers agree on a pair, we assign a probability of 0.95, as opposed to 1.0, to the YES or NO to reflect the fact the this initial evidence is not really perfect.) For across-sports images, their metadata information is sufficient to produce a machine answer of NO with associated probability 1.0.

In order to answer the questions issued by each question selection algorithm, we build a cache of answers from experienced workers with the following AMT qualifications: a) at least 97% of their previous tasks in the platform are approved and b) at least 500 of their previous tasks are approved. For each two same-sport images we collect 10 answers. In the rare case where question selection algorithms end up asking more than 10 questions on a specific pair, we recycle the answers on that pair once we run out of cached answers.

In order to estimate the human accuracy parameter, $p^H_{i,t}$, used by the question selection algorithms, we use a smaller set of 30 images, with 5 “expert” answers per image pair. From these data we compute a $p^H_{i,t}$ of 0.9, i.e., 9 out of 10 answers are correct, on average.

The Gymnastics dataset [12] contains 94 images of gymnastics athletes. The main difference with the AllSports dataset is that Gymnastics contains images where it is very difficult to distinguish the face of the athlete, e.g. the athlete is upside down on the uneven bars.

To construct the initial answers, we follow the same procedure with the one described for AllSports. The only difference is that for each image pair we ask fewer questions, since we need to ask questions for more pairs compared to AllSports (in Gymnastics we ask questions between any two images, as opposed to AllSports where we ask questions only between same-sport images). In particular, we ask for each pair 5 questions to AMT workers without qualifications and 5 questions to workers-“experts”.

Using a training set of 30 images, we compute a human accuracy parameter, $p^H_{i,t}$, of 0.9; the same as in AllSports.
selects the questions to ask next, it will skip questions whose answer can be inferred through transitive relations. However, when SCC is performing the resolution such questions may be very useful. The second reason is that with this dataset MLF99+SCC gets confused by certain “problematic pairs” where humans often provide the wrong answer. (We discuss these problematic pairs more below.) MLF99 repeats each question until it thinks that it has the right answer (with probability at least 0.999). In spite of this repetition, it often gets the incorrect answer for a problematic pair of images, leading to a poor overall clustering.

MLF9999+SCC (not shown in Fig. 9(a) to avoid clutter) repeats each question until it believes it has the correct answer with probability at least 0.9999, reducing the impact of problematic pairs, but consuming more questions in the process. The net result is that MLF9999+SCC starts “slower” than MLF99+SCC, however, after 2500 questions it surpasses MFL99+SCC and stays very close to MLF9999+SCC.

Fig. 9(a) also shows the performance of the solutions that use TC for entity resolution. Initially TC achieves a 0.0 F1 score because it does not use the initial evidence. (With the initial machine evidence, many pairs do not have an above threshold probability, so using transitive closure at that point is not productive [2].) Thus, MLF needs to ask many more questions to improve accuracy. We observe a similar behavior as before: MLF99+TC gets confused by problematic pairs and never goes above an F1 of 0.65; MLF9999+TC performs better but needs to ask more questions due to its repetitions.

The results for the Gymnastics dataset are shown in Fig. 9(b). The Gymnastics dataset is a much tougher one, as it contains significantly more problematic pairs. The net result is that (1) none of the SCC schemes can improve accuracy much beyond the initial F1=0.85 value, and (2) even MLF9999+TC gets confused and flattens out roughly at F1=0.75.

We have argued that one of the key performance factors is the number of problematic pairs. The Gymnastics dataset contains more of them because athletes are often photographed upside down, grimacing or with hair covering their face. Fig. 10 quantifies our notion of problematic pairs for each dataset. The figure shows the percentage of pairs (Y-axis, log scale) that have a given percentage of wrong answers (X-axis). For example, for the Gymnastics set we ask 5 questions for each pair; if we get one correct and 4 wrong answers we have an 80% wrong answer percentage (X-axis), and Fig. 10 tells that about 1% of the pairs were in this category. Note that for the AllSports set where we ask 10 questions, there are 11 possible X-axis values, as opposed to only 6 for Gymnastics.

In addition, note that when the percentage of wrong answers is higher than 0.5, we get an overall wrong answer for the pair if we keep repeating the same question. These pairs are problematic and will cause the algorithms to cluster records incorrectly. Fig. 10 shows that Gymnastics has more problematic pairs; the absolute numbers are not large, but it does not take many problematic pairs to confuse the resolution algorithms. The effect is more pronounced when TC is used for resolution, since a single incorrect pair can cause clusters to join or break up.

2) CAPTCHAs Dataset: The CAPTCHA dataset consists of 244 CAPTCHA images, each showing a four-digit number. The number of records per entity follows a power-law distribution with an exponent of −2.5. We produce initial answers between pairs of images, using a simple image processing algorithm we implemented, which computes the probability distribution for each digit. For example, for the first digit, we compute the probability of the digit being a “0”, the probability of being a “1”, and so on. Combining the distributions for the four digits of two images, we compute an initial answer along with a probability of the answer being correct. Human answers are provided on demand to each question selection algorithm, by a cache of answers we constructed using AMT. In particular, for each of the 29646 pairs we ask 5 questions and store the answers in the cache. In order to explore a scenario with “noisy” human answers we did not set any specific requirements on the workers processing the tasks. Using a smaller training set of CAPTCHAs, we compute a human accuracy, $p^H$, of 0.7. This value is lower than for the images datasets, mainly because we do not use any worker requirements here.

Fig. 11 shows the results for CAPTCHA. There are two differences compared to the results for the images datasets. First, bDENSE needs even fewer questions to converge to a highly accurate outcome. (Note that the total number of pairs we can question is larger than in Gymnastics and AllSports, yet high accuracy is obtained with fewer human questions.) Second, MLF99 converges much faster to high accuracy compared to MLF9999.

There are two factors explaining these differences. First, because of the records/entity power-law distribution used in the CAPTCHA dataset, there are a lot of entities with more than two or three records. It turns out that large entities can “magnify” the impact of question selection. When large entities are correctly identified, the F1 score improves a lot (since many record pairs make up the entity); if mistakes are made, the F1 can significantly suffer. Fig. 11 shows that since bDENSE+SCC asks questions intelligently, with respect to the overall evidence, it quickly finds large entities and the F1 score is boosted. The other algorithms under SCC do not
We start by describing the dataset generation process and the main synthetic data parameters and, then, we discuss our findings for different scenarios. The details of the generation process can be found in the technical report [9].

First, we generate a dataset of REC's records, where the number of records per entity follows a distribution DISTRO. In addition, entities are grouped into buckets. Parameter BUCKET gives the total number of buckets used. The entities in one bucket can have records that look similar, while records are assumed quite dissimilar to those in other buckets.

In particular, the initial evidence for record \( r \) compared to record \( s \) is generated as follows. If \( s \) is in a different bucket than \( r \), the initial evidence is NO (records do not match) with a reported confidence of 1.0. If \( r \) and \( s \) are part of the same entity, we pick a value \( c \) between 0.5 and 1 to represent the confidence of a machine comparison. Then with probability \( c \) the initial evidence is correct (YES), and with probability \( 1-c \) it is incorrect (NO). In either case we associate a confidence of \( c \) with the answer. If \( r \) and \( s \) are in the same bucket but not part of the same entity, with probability \( M_A \) we say that \( r \) and \( s \) are similar, and with probability \( 1-M_A \) we say they are different. If the records are different, then the initial evidence says NO with a confidence of 1.0. If \( r \), \( s \) are similar we again generate a match probability \( c \) between 0.5 and 1, and then report (confidence \( c \)) correct evidence (NO) with probability \( c \) and an incorrect YES with probability \( 1-c \).

For human comparisons, we assume an accuracy \( H_A \): with probability \( H_A \) we report a correct answer (confidence \( H_A \)) and with probability \( 1-H_A \) we report an incorrect answer.

To simulate what we have called problematic pairs, we define a parameter PROBLEMATIC that specifies the fraction of pairs of records of the same entity that are problematic. That is, for each entity we select at random a fraction PROBLEMATIC of its possible pairs and label them as problematic. Then we change the rules (given above) for comparing records \( r \) and \( s \) of the same entity as follows: If \( r \), \( s \) is a problematic pair, flip the answer (i.e., with probability \( c \in [0.5,1.0] \) the initial evidence is incorrect, or with probability \( H_A \) the human answer is incorrect.)

Table I summarizes all the parameters used in the synthetic data experiments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
<th>Base Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC</td>
<td>Total Dataset Records</td>
<td>(900)</td>
<td>500</td>
</tr>
<tr>
<td>DISTRO</td>
<td>Gold Standard Records-Per-Entity Distribution</td>
<td>[GAUSSIAN, POWERLAW]</td>
<td>GAUSSIAN</td>
</tr>
<tr>
<td>BUCKET</td>
<td>Bucketing Factor</td>
<td>(20)</td>
<td>20</td>
</tr>
<tr>
<td>( M_A )</td>
<td>Machine Accuracy</td>
<td>(0.9, 0.6)</td>
<td>0.9</td>
</tr>
<tr>
<td>( H_A )</td>
<td>Human Accuracy</td>
<td>(0.7, 0.9)</td>
<td>0.7</td>
</tr>
<tr>
<td>PROBLEMATIC</td>
<td>Controls problematic pairs</td>
<td>(0.0, 0.2)</td>
<td>0.0</td>
</tr>
</tbody>
</table>

In order to generate each curve in the plots in this section, we generate 3 different gold clusterings, and for each clustering we repeat the experiment 4 times. The curve for each approach is the average over the 12 runs.

For the base case in Table I, Fig. 12(a) gives the F1 score, as different approaches issue new questions, until we reach a limit of 10000 questions. Note that in the base case we use a Gaussian distribution for the number of records per entity (DISTRO), with a mean of 3.0 and a variance of 2.0.
Next, we examine the effect of lowering the quality of the initial answers. Hence, we decrease the machine accuracy, $M_A$, from 0.9 to 0.6. Human accuracy, $H_A$, is still 0.9 and the rest of the parameters remain the same. As we see in Fig. 14(a) bDENSE+SCC once again outperforms the other approaches, requiring 40% fewer questions to reach $F_1 = 0.9$, compared to the second best, MLF99+SCC. Even with a machine accuracy of 0.6, there is enough information in the initial evidence to let bDENSE select reasonable initial human questions, and since human answers are high quality, the $F_1$ score improves rapidly.

In order to create a scenario where bDENSE does worse than some of the other approaches, we need to change two additional parameters. First we simulate problematic pairs by setting PROBLEMATIC to 20%. This means that in 20% of the cases where we ask a human to compare records that are part of the same entity, they will give us the wrong answer with high probability ($H_A = 0.9$). (The initial evidence will be similarly biased.) Second, we use a POWERLAW DISTRO with an exponent of $-3.0$. This distribution gives us some large entities, which as we discussed earlier amplify the impact of question selection. The rest of the parameters remain the same as in the base case, except for human accuracy which is 0.9 as in our previous experiment.

As we see, in Fig. 14(b), the performance of all four approaches drops significantly. The effect is more marked with bDENSE+SCC. In this situation, bDENSE’s attempt to select questions to ask intelligently backfires since it is based on incorrect evidence. And this poor performance is magnified when large entities are incorrectly resolved. On the other hand, HALF, does not take into account the graph structure, and ends up asking relatively better questions. Similarly, even MLF99+TC eventually achieves a better $F_1$ value by being conservative and asking and re-asking questions that already have relatively good evidence.

The scenario of Fig. 14(b) simulates and explains the behavior we observed with some real datasets, when the quality of initial answers was poor. To example, in the Cora dataset we find quite a few entities that have almost identical fields and also has large entities. For instance, one entity may have say 10 records, each representing the same conference paper. Another entity may have say 40 records each representing the same conference paper. A similarity function that heavily relies on the title and set of authors, will generate YES answers with high confidence between any of the 10+40 records; since those records have the same title and set of authors. Thus, bDENSE will not detect a “lack”-of-evidence between those records and defer any questions on...

---

Fig. 13. Simulator screenshots comparing MLF and bDENSE question selection.

The reasons why bDENSE performs well compared to the other strategies were discussed in Section VI-A. Now with our simulator we can confirm those observations by seeing how particular sets of records are resolved. To illustrate, Fig. 13 shows how 4 given entities are resolved. In this case, the generator created one entity with 4 records, one entity with 3 records, two entities with 2 records, and one entity with a single record. On the left side of the figure, we see the initial answers, used as input by all approaches, along with the generated clustering. We use green dots for records, dashed red lines for NOs, and solid blue lines for YESes. The thickness of each line is proportional to the probability of the combined answer for the two corresponding records. In the middle, we see how MLF chooses to spend 20 questions, while on the right side, we see how bDENSE spends 20 questions. MLF chooses to “fully” resolve the relation between 5 pairs of records, while bDENSE distributes the 20 questions on 11 pairs of records. Instead of trying to build large connected components like MLF, bDENSE does not insist on certain pairs of records and prioritizes pairs based on the “lack”-of-evidence it detects. For example, note that inside the entity of 4 records, bDENSE asks questions on 3 record pairs with “weak” initial answers (thin edges in the left side of the figure), while MLF asks questions on 3 record pairs with “strong” initial answers (thick edges in the left side of the figure).

In Fig. 12(b) we examine a scenario with more “reliable” workers. Thus, we keep the same parameters with the base case, however, we change human accuracy, $H_A$, from 0.7 to 0.9. All five approaches converge much faster compared to the base case of Fig. 12(a). This effect is analogous to what we observed in Fig. 11 (CAPTCHAS dataset). Again, bDENSE+SCC is considerably better than the other approaches, and needs 75% fewer questions than MLF99+SCC to reach 0.95, and 85% fewer questions than HALF99+SCC.
this part of the dataset.

VII. RELATED WORK

Entity Resolution is a well-studied problem and goes by various names including record linkage, deduplication, identity resolution, object identification, merge/merge, reference reconciliation. Surveys [15] and [16] discuss approaches and algorithms proposed for Entity Resolution.

Recently, a number of approaches have been proposed for enhancing ER with evidence provided by humans [1]. [2], [3], [4], [5], [6], [7]. Reference [18] proposes an interface where a human sees more than two records and tags records representing the same entity with the same label. The main objective in [18] is the “efficient packaging” of records into tasks for humans. A similar interface with more than two records in each human task is used in [17]. The main focus of [17] is the probabilistic aggregation of human tasks’ outcome into an overall clustering.

As opposed to [17] and [18], references [1], [2], [3], [4] use a simpler interface for the human task, involving only two records. All of [1], [2] and [3], propose question selection algorithms for ER, having as their main assumption that humans do not make errors. On the contrary, our approach assumes that humans are not always correct. The same assumption is also made by [7], however, the main focus of [7] is designing an Entity Resolution Algorithm (ERA) that can handle human errors. The question selection process of [7] just randomly selects the record pair to resolve at each step; once the two records are selected the paths connecting them are examined to find the question on the edge that will reveal the “most” about the relationship of the records.

An approach complementary to our approach is proposed in [19], where crowdsourcing is applied in additional steps of an ER workflow, e.g., by using active learning to train the similarity function producing the initial (machine) answers.

References [20], [21], [22] study similar problems to the question selection for ER. Reference [20] defines the “most informative” question as the question that will cause the “biggest” change in the current clustering, after the answer for that question is retrieved. References [21] and [22] focus on spectral clustering [23], and try to find the questions that will enhance the “most” the evidence used for spectral clustering. While spectral clustering can be used for a k-way partitioning of records into clusters, the approaches proposed in [21] and [22] focus in the 2-way partitioning case. As opposed to [20], [21], [22], our definition for the “most beneficial” question, along with our results and algorithm, are based on a Maximum Likelihood formulation.

The Maximum Likelihood clustering is closely related to the problem of correlation clustering [10]. In fact, the ML clustering, presented in Section II-B can be reduced to the weighted version of correlation clustering, as [24] discusses.

VIII. CONCLUSION

We studied the problem of enhancing Entity Resolution using human answers that may possibly be erroneous. The key challenge is how to reason about likely record clusters (entities) when both the computer generated and human generated evidence can be incorrect. We derived theoretical results that help us identify beneficial questions to ask humans, i.e., questions whose answers can improve the accuracy of the Maximum Likelihood clustering. These insights lead to a practical and efficient algorithm (bDENSE) for selecting questions to ask. Through our experiments we showed that this algorithm often outperforms other approaches, even when heuristic clustering algorithms (ERAs) are used to perform the final record partition.

We also discovered that ERAs that apply simple thresholds do not handle erroneous evidence well. Instead we suggested a Spectral Connected Components (SCC) ERA that only merges two clusters when the overall evidence indicates that it is likely that the two sets of records represent the same entity. Our experiments indicate that this SCC ERA outperforms other approaches (unless computer or human evidence is very misleading).

REFERENCES