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.

1. 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. 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 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. 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. 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 [18, 20]) 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 [12, 13, 16] 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 [11] 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 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 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 2.3.

• We provide theoretical results that identify “beneficial” questions with a reasonable computational cost, in Section 3.

• We propose a lightweight algorithm that applies our theoretical results to select questions, in Section 4.

• We present our SCC clustering algorithm for partitioning records based on probabilistic evidence, in Section 5.

• We compare our question selection algorithm to two algorithms [18] [20] proposed in related work, using datasets of images and AMT [2] workers, and we identify in which cases each algorithm is better, in Section 6.

1.1 Running Example

Consider the dataset of 4 records in Fig. 1. 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 (b, 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.

2. PRELIMINARIES

2.1 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 refer to different entities. Note the subtle difference between this model and the model of [20]. To illustrate, consider three records a, b, and c that refer to a single entity. In [20], 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 \( p^H \) error rate assumption to simplify the question selection reasoning. In our experimental evaluation we show that in datasets 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 \( P \).

In the machine answer case, the similarity-to-probability process must take into account the amount of information used for generating the similarity value. For example, consider two records, a and b, with 3 fields, full-name, phone-number, and country. If a and b have identical values in all 3 fields, they will have a max similarity value of 1.0. On the other hand, two records, c and d, missing values on full-name and phone-number, and having the same country value will also have similarity 1.0. However, the probability of referring to the same entity for c and d should be significantly lower than a and b. Moreover, if two other records, c and f, were missing values in full-name and country and had the same phone-number value, they should receive a higher probability than c and d, because the phone number is more indicative than country. Note also that how indicative some fields are, depends on the specific dataset. For example, fields brand and type in a dataset containing only Apple laptops are not indicative at all. In Section 6.1.3, we will examine how a low quality similarity-to-probability translation can affect our approach.

2.2 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. 1, $E$ consists of five answers. First, we will examine the most probable clustering of records in Fig. 1 based on $E$, such that all records of each formed cluster refer to one entity. An example of such a clustering appears in Fig. 2(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

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

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 Section 5, 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. 2(a). $P(E|C_1) = 0.8^4 \times 0.6 \times (1 - 0.6)^2$, i.e., answers about the pairs $(a, b)$, $(c, d)$, $(a, d)$ and $(b, c)$ are proven to be correct, while answers for the pairs $(a, d)$ and $(b, c)$ are proven to be false, if $C_1$ is the “true” clustering.

![Figure 2: Clusterings C1 and C2](image)

A clustering with a higher likelihood $P(E|C_k)$ is clustering $C_2 = \{(a, b), (c, d)\}$, depicted in Fig. 2(b). $P(E|C_2) = 0.8^2 \times 0.6^2 \times (1 - 0.6)^2$, 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. 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 correlation clustering [4]. In correlation clustering the objective is to find a clustering that minimizes the number of “disagreements”. For instance, in Fig. 2(b) the YES between records $b$ and $d$ forms a disagreement for clustering $C_2$. In this paper, however, we are interested in addressing a different problem; finding the questions to ask next. In the next section, we provide a formal definition for the simple case of this problem, where we are interested in finding the next single question to ask next.

2.3 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 3 that form the basis of our algorithms, described in Section 4.

Let us examine which would be the “best” single 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:

$$P(y_{ij}|E) \max_k P(C_k|E \land y_{ij}) = \frac{P(E \land y_{ij})}{P(E)} \max_k \frac{P(E \land y_{ij}|C_k)P(C_k)}{P(E \land y_{ij})} = \frac{1}{P(E)} \max_k P(E \land y_{ij}|C_k)P(C_k)$$

Likewise, in case of a NO we have:

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

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 $\frac{P_C}{P(E)}$.

3. THEORETICAL RESULTS

We start with some preliminary results and definitions that will simplify the statements and proofs of the main results, in THEOREM 1 and COROLLARIES 1, 2, and 3. The main results will be later applied by our algorithm in Section 4. 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.

**Lemma 1.** For any question $q_{ij}$, $\alpha(ij) \in [\mathcal{ML}(E), 2p_C^H \mathcal{ML}(E)]$

**Proof.** First, note that since human participants are not malicious, and $p_C^H \geq 0.5$, it follows that $2p_C^H \mathcal{ML}(E) \geq \mathcal{ML}(E)$. Consider an arbitrary question $q_{ij}$. Next, we examine the best and worst case scenarios:

1. YES answer for $q_{ij}$
Now we examine the overall worst and best case scenarios:

(a) **Best case scenario:** There is an ML clustering, $C_{ML}$ for $E$, that has records $i$ and $j$ in the same cluster. In that case the YES answer we got back is considered correct for $C_{ML}$. Therefore,

$$\mathcal{M}(E \land y_{ij}) = p^H_c \mathcal{M}(E)$$

(b) **Worst case scenario:** Any ML clustering for $E$ has records $i$ and $j$ in two different clusters. For $E \land y_{ij}$, there are two cases for the ML clustering:

i. Any ML clustering for $E \land y_{ij}$ has $i$ and $j$ in two different clusters. In this case, the new YES answer is considered wrong and:

$$\mathcal{M}(E \land y_{ij}) = (1 - p^H_c)\mathcal{M}(E)$$

ii. There is an ML clustering for $E \land y_{ij}$ with $i$ and $j$ in the same cluster. Consider $C$, an ML clustering for $E$. $P(E \land y_{ij}|C) = (1 - p^H_c)\mathcal{M}(E)$, since any ML clustering for $E$ has records $i$ and $j$ in two different clusters.

If $C$ is an ML clustering for $E \land y_{ij}$ then,

$$\mathcal{M}(E \land y_{ij}) = (1 - p^H_c)\mathcal{M}(E)$$

Otherwise,

$$\mathcal{M}(E \land y_{ij}) > (1 - p^H_c)\mathcal{M}(E)$$

2. **NO answer for $y_{ij}$**

(a) **Best case scenario:** There is an ML clustering, $C_{ML}$ for $E$, that has records $i$ and $j$ in different clusters. Thus, the NO answer is considered correct for $C_{ML}$ and

$$\mathcal{M}(E \land n_{ij}) = p^H_c \mathcal{M}(E)$$

(b) **Worst case scenario:** Any ML clustering for $E$ has records $i$ and $j$ in the same cluster. For $E \land n_{ij}$, there are two cases for the ML clustering:

i. Any ML clustering for $E \land n_{ij}$ has $i$ and $j$ in the same cluster. In this case, the new NO answer is considered wrong and:

$$\mathcal{M}(E \land n_{ij}) = (1 - p^H_c)\mathcal{M}(E)$$

ii. There is an ML clustering for $E \land n_{ij}$ with $i$ and $j$ in two different clusters. Consider $C$, an ML clustering for $E$. $P(E \land n_{ij}|C) = (1 - p^H_c)\mathcal{M}(E)$, since any ML clustering for $E$ has records $i$ and $j$ in the same cluster. If $C$ is an ML clustering for $E \land n_{ij}$ then,

$$\mathcal{M}(E \land n_{ij}) = (1 - p^H_c)\mathcal{M}(E)$$

Otherwise,

$$\mathcal{M}(E \land n_{ij}) > (1 - p^H_c)\mathcal{M}(E)$$

Now we examine the overall worst and best case scenarios:

- **Overall best case scenario:** The upper bound for $\alpha(ij)$ is achieved when the best case scenarios (1a) and (2a) happen. This is possible, when there are at least two ML clusterings for $E$, one having records $i$ and $j$ in the same cluster and one having the two records in different clusters. In that case,

$$\alpha(ij) = \mathcal{M}(E \land y_{ij}) + \mathcal{M}(E \land n_{ij}) = 2p^H_c \mathcal{M}(E)$$

- **Overall worst case scenario:** For the lower bound, note that when the best case scenario (1a) does not happen, then the best case scenario (2a) must hold. If (1a) does not hold then there will be an ML clustering for $E$ with records $i$ and $j$ in different clusters, and, therefore, case (2a) will hold. In a similar fashion, if case (2a) does not hold, (1a) must hold. In the worst case, either cases (1b) and (2a) hold, or (1a) and (2b) hold. Hence,

$$\alpha(ij) = \mathcal{M}(E \land y_{ij}) + \mathcal{M}(E \land n_{ij}) = (p^H_c + 1 - p^H_c)\mathcal{M}(E) = \mathcal{M}(E)$$

An example where the upper bound is achieved for $\alpha(ij)$ is given in Fig. 3. Initial evidence $E$ consists of four answers and there are two ML clusterings for $E$, $C_1 = \{(a,b), (c,d)\}$ and $C_2 = \{(a,b,c,d)\}$. Consider images $a$ and $c$. Both of the best case scenarios, (1a) and (2a), take place, i.e., in $C_1$, $a$ and $c$ are in different clusters, while in $C_2$, $a$ and $c$ are in the same cluster. Thus, question $q_{ac}$ achieves the upper bound, $2p^H_c \mathcal{M}(E)$. This is also the case for questions $q_{bd}$, $q_{ac}$, and $q_{ad}$.

![Figure 3: Upper bound achieved by $q_{ac}, q_{bd}, q_{ac}$, and $q_{ad}$](image)

Intuitively, a question on the lower bound, i.e., with $\alpha = \mathcal{M}(E)$, does not increase our confidence for the ML clustering. On the other hand, any question with $\alpha > \mathcal{M}(E)$, 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, $2p^H_c \mathcal{M}(E)$, 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 $\mathcal{M}(E)$, is a beneficial question.

**DEFINITION 2.** **Optimal question**: A question with an $\alpha$ equal to $2p^H_c \mathcal{M}(E)$, is an optimal question.

For instance, $q_{ac}$ is a beneficial question in our running example of Fig. 3. A NO gives $C_2$ as the ML clustering, as shown in Fig. 4(a) with

$$\mathcal{M}(E \land n_{ac}) = p^H_c P(E|C_2) = p^H_c \mathcal{M}(E)$$

A YES gives $C_1$ as the ML clustering, as shown in Fig. 4(b)

with
Clustering, two clusters as since $P(C_1) = 0.8^2 \times 0.6 \times (1 - 0.6)^2$ and $\mathcal{ML}(E) = 0.8^2 \times 0.6^2 \times (1 - 0.6)$ Thus,

$$\alpha(ac) = p_c^N (1 + \frac{0.5}{0.6^2 \times (1 - 0.6)}) \mathcal{ML}(E) \simeq 1.33 \mathcal{ML}(E)$$

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

$$\alpha(ab) = \mathcal{ML}(E \cup y_{ij}) + \mathcal{ML}(E \cup n_{ij}) = p_c^N \mathcal{ML}(E) + (1 - p_c^N) \mathcal{ML}(E) = \mathcal{ML}(E)$$

The intuition, in this example, is that while records $(a, b)$ seem to refer to one entity and records $(c, d)$ also seem to refer to one entity, it is not clear if $(a, b)$ refer to a different entity than $(c, d)$, or if all four records refer to one single entity. Thus, asking $q_{ac}$ is more “usefull” than asking $q_{ab}$. This notion of “uncertainty” is formally defined by $\lambda$-balance.

$\lambda$-balance expresses how “uncertain” $Y$ are we 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. 5(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 $p_c^N(a)$. We denote:

$$P_Y = \prod_{a \in Y} p_c^N(a), \quad P_Y' = \prod_{a \in Y} (1 - p_c^N(a))$$

$$P_N = \prod_{a \in N} p_c^N(a), \quad P_N' = \prod_{a \in N} (1 - p_c^N(a))$$

and we say that there is a $\lambda$-balance between $c_1$ and $c_2$, for

$$\lambda = \frac{P_N \times P_Y'}{P_N' \times P_Y}.$$ 

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

**Proof.** Let us denote the ML clustering by $C_A$ and the two clusters as $c_1$ and $c_2$. If the overall evidence is the set of answers $E$, we have:

$$\mathcal{ML}(E) = P(E \cup \{Y \cup N\})|C_A \times P_N \times P_Y'$$

Now, consider a clustering $C_A'$ that is identical to $C_A$, with the only difference of having clusters $c_1$ and $c_2$ merged into one single cluster. For $C_A'$, we have $P(E \cup \{Y \cup N\})|C_A' = P(E \cup \{Y \cup N\})|C_A$, and, thereafter,

$$P(E|C_A') = P(E \cup \{Y \cup N\})|C_A' \times P_N \times P_Y'$$

Because $C_A$ is an ML clustering, $\mathcal{MC}(E) \geq P(E|C_A')$. Hence, $P_N \times P_Y' \geq P_N' \times P_Y$, and,

$$\lambda = \frac{P_N \times P_Y'}{P_N' \times P_Y} \leq 1.0$$

In addition, $\lambda \geq 0.0$, because probability is always non-negative.

**Figure 5. $\lambda$-balance description**

In the ML clustering of our running example, in Fig. 2(b) between the two clusters $\{a, b\}, \{c, d\}$, we have $N = \{n_{ad}, n_{bc}\}$, $Y = \{y_{ad}\}$, $P_N = 0.6^2$, $P_N' = (1 - 0.6)^2$, $P_Y = 0.6$ and $P_Y' = (1 - 0.6)$. A $\lambda$-balance exists between the two clusters, for $\lambda = \frac{(1-0.6)^2 + 0.6}{2} = \frac{2}{3}$. In case the probability of a correct answer from a machine, $p_c^M$, for the three machine answers, $n_{ad}, n_{bc}$, and $y_{ad}$, was lower than 0.6, then the $\lambda$-balance between $\{a, b\}$ and $\{c, d\}$, would be higher than $\frac{2}{3}$. On the other hand, if $p_c^M$ was higher than 0.6, $\lambda$ would be lower than $\frac{2}{3}$. In Fig. 5(b) we plot the $\lambda$-balance between $\{a, b\}$ and $\{c, d\}$, as $p_c^M$ 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 4, 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 $\lambda > \frac{1-p_c^H}{p_c^M}$, any question between a record $i$ from the first cluster and a record $j$ from the second cluster, is a beneficial question. In addition, $\alpha(ij) \geq (1 + \lambda)p_c^H \mathcal{ML}(E)$.

**Proof.** Let $C_{ML}$ be the ML clustering, $c_1$ and $c_2$ the two clusters and $q_{ij}$ the arbitrary question we ask, where $i$ belongs to $c_1$ and $j$ to $c_2$. 

(a) $\lambda$-balance definition for two (b) $\lambda$-balance as $p_c^M$ clusters

P goes from 0.5 to 1.0, in the running example
1. NO answer for \( q_{ij} \):
\[
\mathcal{M}(E \cap n_{ij}) = p^H_c \mathcal{M}(E)
\]
since the ML clustering for \( E \) is also an ML clustering for \( E \cap n_{ij} \).

2. YES answer for \( q_{ij} \): Consider the clustering \( C_{\text{merge}} \) which is the exact same clustering with \( C_{\text{ML}} \), except that we merge \( c_1 \) and \( c_2 \) into one cluster. The likelihood of \( C_{\text{merge}} \) for our initial evidence \( E \), is \( \lambda \mathcal{M}(E) \). For \( E \cap n_{ij} \) the likelihood of \( C_{\text{merge}} \) becomes \( p^H_c \lambda \mathcal{M}(E) \). Thus, the ML clustering for \( E \cap y_{ij} \) has a likelihood of at least \( p^H_c \lambda \mathcal{M}(E) \), and
\[
\mathcal{M}(E \cap y_{ij}) \geq p^H_c \lambda \mathcal{M}(E)
\]
For question \( q_{ij} \),
\[
\alpha(ij) = \mathcal{M}(E \cap n_{ij}) + \mathcal{M}(E \cap y_{ij}) \geq (1 + \lambda)p^H_c \mathcal{M}(E)
\]
Moreover, since \( \lambda > \frac{1 - e^{-1/2}}{e^{-1/2}} \), \( \alpha(ij) > (1 + \frac{1 - e^{-1/2}}{e^{-1/2}})p^H \mathcal{M}(E) = \mathcal{M}(E) \). Therefore, \( q_{ij} \) is a beneficial question.

An application of THEOREM 1 appears in the running example, in Fig. 2(b) where there is a \( \lambda \)-balance between the two clusters, \( \{a, b\} \), \( \{c, d\} \), of the ML clustering, for \( \lambda = \frac{(1-0.6)^2+0.6}{0.6+0.4} \). Questions \( q_{ac}, q_{ad}, q_{bc}, q_{bd} \) are beneficial with an \( \alpha \) of, at least, \( (1 + \lambda)0.8 \mathcal{M}(E) \approx 1.33 \mathcal{M}(E) \); 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 from one connected component and any record from another connected component, is an optimal question.

**PROOF.** Consider the two connected components, \( S_i \) and \( S_j \), and pick arbitrarily a record \( i \) from the first and a record \( j \) from the second. This situation is depicted in Fig. 2(a) the “Rest of the Graph” can be the empty set. Moreover, consider an ML clustering \( C_{\text{ML}} \), for the evidence \( E \). There are two cases:

1. In \( C_{\text{ML}} \), \( i \) and \( j \) belong to the same cluster. Let us denote this cluster by \( c \) and remove some of the elements from it, in order to construct two new clusters. Specifically, we construct clusters \( c' \equiv e \cap S_i \) and \( c'' \equiv e \cap S_j \). Moreover, by removing elements from \( e \) we construct a third new cluster, \( c \equiv c \setminus \{c', c''\} \). Since \( S_i \) and \( S_j \) are two separate connected components, there are no answers between \( c' \) and \( c'' \), between \( c' \) and \( c \), and between \( c \) and \( c'' \). Therefore, the new clustering does not “violate” any answer and it is also an ML clustering for \( E \). In addition, in the new clustering there is a 1.0–balance between clusters \( c' \) and \( c'' \), since no question has been asked between their elements. Based on THEOREM 1, \( \alpha(ij) \geq (1 + 1)p^H_c \mathcal{M}(E) \) and, therefore, \( q_{ij} \) is an optimal question.

2. In \( C_{\text{ML}} \), \( i \) and \( j \) belong to different clusters. Let us denote these clusters by \( c_i \) for record \( i \) and \( c_j \) for record \( j \). We apply the same construction process as in case (1), and we form two new clusters \( c' \equiv e \cap S_i \) and \( c'' \equiv e \cap S_j \). Again, in the new clustering there is a 1.0–balance between clusters \( c' \) and \( c'' \), and, based on THEOREM 1, \( q_{ij} \) is an optimal question.

**Figure 6:** The cases where COROLLARY 1 and COROLLARY 2 applies

Fig. 6(a) depicts the case where COROLLARY 1 applies. \( S_i \) and \( S_j \) are two connected components of \( \mathcal{G} \), i.e., there are no answers between \( S_i \) and \( S_j \) or between \( S_i \) and \( S_j \) and the “Rest of the Graph”. As COROLLARY 1 states, a question between a record \( i \) from \( S_i \) and a record \( j \) from \( S_j \) is an optimal question. 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. 6(b).

**COROLLARY 2.** Consider two connected components of \( \mathcal{G} \) 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.

**PROOF.** Consider two connected components, \( S_i \) and \( S_j \), for which COROLLARY 2 applies, and two arbitrarily picked records \( i \) and \( j \) from the two components. The case is depicted in Fig. 6(b). We can follow the exact same construction process as in the proof of COROLLARY 1 and get an ML clustering with two new clusters \( c'_i \) and \( c'_j \). Again, between \( c'_i \) and \( c'_j \) there is a 1.0–balance and \( q_{ij} \) 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 4, we discuss in detail the complete algorithm that uses the corollaries presented here.

Now, let us consider a more general case, shown in Fig. 7(a). The situation is the same with the one depicted in Fig. 6(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 $a(i,j)$? Such a guarantee would help us find beneficial questions. As Corollary 3 states, $q_{ij}$ is a beneficial question, with $a(i,j) \geq (1 + \frac{4^2}{3^2})p_c^H \mathcal{M}(E)$.

Next, we give the general definition for the ratio $\frac{4^2}{3^2}$ resulting from answers a) and b), in this example.

(a) Motivation example

(b) General case where Corollary 3 applies

**Figure 7: Corollary 3 description**

**Definition 4. $p$-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 $R \setminus \{S_i \cup S_j\}$ by $Y_1$, b) the YES answers between $S_j$ and $R \setminus \{S_i \cup S_j\}$ 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) $P_Y = \prod_{a \in Y} p_c(a)$, 2) $P_Y^2 = \prod_{a \in Y} p_c(a)$, 3) $P_N = \prod_{a \in N} p_c(a)$, 4) $P_Y = \prod_{a \in Y} p_c(a)$, 5) $P_Y = \prod_{a \in Y} (1 - p_c(a))$, 6) $P_Y^2 = \prod_{a \in Y} (1 - p_c(a))$, 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 $p$-ratio, the ratio:

$$\rho = \frac{P_Y + P_Y^2}{P_Y \times P_Y^2} \times \min(P_N, \frac{P_Y}{P_N})$$

**Corollary 3.** When between two disjoint subsets of records, $S_i$ and $S_j$, there is a $p$-ratio with

$$\rho > (1 - p_c^H)^p$$

a question between an arbitrary record $i$ of $S_i$ and an arbitrary record $j$ of $S_j$ is a beneficial question.

Moreover,

$$a(ij) \geq (1 + \rho) p_c^H \mathcal{M}(E)$$

**Proof.** Consider two arbitrarily picked records $i$ and $j$, from $S_i$ and $S_j$, as they appear in Fig. 7(b). Let $c_i$ be the cluster of $i$ and $c_j$ the cluster of $j$, in an ML clustering, $\mathcal{C}_{ML}$. Next, we are going to describe the worst case scenario when (1) clusters $c_i$ and $c_j$ are different clusters and (2) clusters $c_i$ and $c_j$ are the same cluster.

1. $c_i$ and $c_j$ are different clusters

   (a) **YES answer for $q_{ij}$:** The ML clustering for $E \setminus n_{ij}$ is $\mathcal{C}_{ML}$, with likelihood equal to $p_c^H \mathcal{M}(E)$.

   (b) **YES answer for $q_{ij}$:** In this case, the likelihood of the ML clustering for $E \setminus y_{ij}$ may be a lot less than $p_c^H \mathcal{M}(E)$. We can get a lower bound on $\mathcal{M}(E \setminus y_{ij})$, by considering the following clustering. Remove from $c_i$ the subset $c_i \cap S_i$, remove from $c_j$ the subset $c_j \cap S_j$, and create a new cluster $\{c_i \cap S_i \cup c_j \cap S_j\}$. The likelihood of the constructed clustering for $E \setminus y_{ij}$ is, at least

$$\frac{P_Y \times P_Y^2 + P_N}{P_Y \times P_Y^2 + P_N} p_c^H \mathcal{M}(E)$$

because we can not “violate” any additional answers except for those in $\{Y_1 \cup Y_2 \cup N\}$.

Hence,

$$\mathcal{M}(E \setminus y_{ij}) \geq \frac{P_Y \times P_Y^2 + P_N}{P_Y \times P_Y^2 + P_N} p_c^H \mathcal{M}(E)$$

2. $c_i$ and $c_j$ are the same cluster $c$

   (a) **YES answer for $q_{ij}$:** The ML clustering for $E \setminus y_{ij}$ is $\mathcal{C}_{ML}$, with likelihood equal to $p_c^H \mathcal{M}(E)$.

   (b) **NO answer for $q_{ij}$:** We follow a similar construction process as in (1b) to provide a lower bound on $\mathcal{M}(E \setminus n_{ij})$. We consider the following clustering. Remove from $c$ the subset $c \cap S_i$, remove from $c$ the subset $c \cap S_j$, and create two new clusters $\{c \cap S_i\}$ and $\{c \cap S_j\}$. The likelihood of the constructed clustering for $E \setminus n_{ij}$ is, at least

$$\frac{P_Y \times P_Y^2 + P_N}{P_Y \times P_Y^2 + P_N} p_c^H \mathcal{M}(E)$$

because we can not “violate” any additional answers except for those in $\{Y_1 \cup Y_2 \cup Y\}$.

Hence,

$$\mathcal{M}(E \setminus y_{ij}) \geq \frac{P_Y \times P_Y^2 + P_N}{P_Y \times P_Y^2 + P_N} p_c^H \mathcal{M}(E)$$

By combining cases (1) and (2), we get the overall worst case scenario, where we have:

$$a(ij) = \mathcal{M}(E \setminus y_{ij}) + \mathcal{M}(E \setminus n_{ij}) \geq \left(1 + \frac{P_Y \times P_Y^2 + \min(P_N, \frac{P_Y}{P_N})}{P_Y \times P_Y^2 + \min(P_N, \frac{P_Y}{P_N})} p_c^H \mathcal{M}(E)\right)$$

Since

$$\rho = \frac{P_Y \times P_Y^2 + \min(P_N, \frac{P_Y}{P_N})}{P_Y \times P_Y^2 + \min(P_N, \frac{P_Y}{P_N})} > (1 - p_c^H)$$

we have:

$$a(ij) \geq (1 + \rho) p_c^H \mathcal{M}(E) \geq (1 + (1 - p_c^H) p_c^H \mathcal{M}(E)) = \mathcal{M}(E)$$

Therefore, $q_{ij}$ is a beneficial question. \qed

4. **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.

While the heuristic we present in this section does not always find an optimal or even a beneficial question (in 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 3, it is not aligned with the 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 $\left(1-e^2\right)$ 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 3 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: **question**: the next question to ask

2: Pick a pair of records with the highest $\rho$-ratio between them

3: **candidates** =

4: while **candidates** is not empty do

5: Pick the answer with the highest YES probability between records in different sets

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). The time complexity of Alg. 1 is $O(n^2)$, where $n$ is the number of records in the dataset.

The rationale behind Alg. 1 is that by merging record sets with strong YES evidence between them, we are less likely to miss a pair of record sets with a high $\rho$-ratio between them. Our choice is related to the fact that $\rho$-ratio is low when YES answers with a high probability are involved. Therefore, we merge sets with "strong" YES answers between them, at each step, so that these "strong" YES answers are not involved in subsequent $\rho$-ratios examined by the algorithm.

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 6, we will use bDENSE, the batch version of our heuristic.

5. 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 5) 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 6 we show that our questions are indeed useful even when the heuristic algorithms are used.

5.1 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 of records, 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 E1 consists of 5 records and entity E2 also consists of 5 records. Therefore, there are 25 pairs of records with one record from E1 and one record from E2. 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 E1 and E2. As the number of constituent records increase, the chances that one record pair causes an erroneous merge increase.

The SCC ERA avoids the problem of Fig. 8(a) by only merging clusters like E1 and E2 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 E1 and E2 being a different entity (event dif), given the 4 answers between E1 and E2, denoted by A. Here, 
P(dif|A) = \frac{P(A|dif)P(dif)}{P(A|same)P(same) + P(A|dif)P(dif)} = \frac{0.3 \times 0.4 \times 0.4 \times 0.9}{0.7 \times 0.6 \times 0.6 \times 0.1 + 0.3 \times 0.4 \times 0.4 \times 0.9} \approx 0.63

SCC assumes that P(same) = P(dif), i.e., answers involving records that do not belong to E1 or E2 are not taken into account for the merging decision. In each step, SCC finds the pair of clusters with the highest P(same|A). If P(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(same|A).

6. EXPERIMENTAL RESULTS

We compare the batch version of our heuristic, bDENSE, with two alternative approaches proposed in [18] and [20]. First, we give a brief overview of the algorithms used for entity resolution, the metric used to measure the performance of the three approaches, and the algorithms for question selection. Then, we describe, in detail, the settings in each experiment and our findings.

Entity Resolution Algorithms (ERA): We use TC and SCC, the two ERAs discussed in Section 5, in our experiments. In particular, TC is used in case the question selection algorithm of [18] is applied, and SCC is used in all other cases. The TC threshold we use is the same with the one used by the question selection algorithm of [18]. Later, in Question Selection Algorithms, we describe the approach of [18] in detail, and we describe the resolution threshold for a pair and how exactly the transitive relation is applied.

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 either the SCC resolution algorithm or the TC algorithm; depending on the question selection method. Then, we compare the output of SCC or TC, to the gold standard clustering. Specifically, we compare the pairs of records that refer to the same entity in the SCC or TC output, to the pairs of records referring to the same entity in the gold standard. We get the precision(p) and recall(r) by comparing the two sets of record pairs and we compute the F1 score, \(F1 = \frac{2 \times p \times r}{p + r}\).
Question Selection Algorithms: We compare bDENSE, the batch version of our heuristic described in Alg. 3 with HALF, the heuristic proposed in [20], and MLF the Maximum-Likelihood-First strategy proposed in [18].

HALF picks in each step the most uncertain pair of records, i.e., the pair of records 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. On the other hand, if for b and d we have a NO answer, then we can infer for pairs, a-d, and c-d, that they refer to different entities.

Here, we should mention that in [18] and [20], the assumption for the HALF and MLF approaches is that humans do not make errors. Therefore, since the human answers we have in our experiments contain errors, for each pair of records HALF and MLF select, we keep asking humans questions until the pair is fully resolved. For example, consider a record pair, (a, b), with a NO answer, with probability of being correct 0.6, and a YES answer with probability of being correct 0.8. The probability of the two records referring to different entities, given the two answers, P(\text{diff}|y_{ab} \land n_{ab}) =

\[
P(\text{diff}|y_{ab} \land n_{ab}) \approx 0.2727
\]

Note that we assume equal priors, i.e., P(\text{same}) = P(\text{diff}). Likewise, P(\text{same}|y_{ab} \land n_{ab}) \approx 0.7273. If we use a threshold of 0.99 for the “full resolution” of a pair, we will keep asking questions on (a, b), until P(\text{diff}|A_{ab}) \geq 0.99 or P(\text{same}|A_{ab}) \geq 0.99, where A_{ab} is the set of answers collected for (a, b). When MLF is combined with TC, TC uses as threshold the threshold MLF uses for the “full resolution” of a pair. We use two different thresholds for pair resolution, 0.99 and 0.9999. The corresponding algorithms are HALFDENSE, HALF9999, MLF9999, and MLF99999.

6.1 Real Data

We use two datasets of images, AllSports [1] and Gymnastics [4], where each image shows an athlete. We use Amazon Mechanical Turk (AMT [2]) to get answers from humans for every pair of records. By using AMT’s qualification mechanisms for workers, we classify answers into a) low-quality answers, from workers without specific qualifications and b) high-quality answers from workers with high accuracy statistics. An initial answer for a pair of records, is formed using the low-quality answers, and the subsequent answers for questions issued by the question selection algorithms are provided, on demand, by the high-quality pool.

In addition, we discuss our results with a third dataset, Cora [3], in Section 6.1.3.

6.1.1 AllSports

AllSports [1] dataset contains 267 athlete images from 10 different sports. Relying on metadata information that indicates which sport is being played in each image, we group the images into 10 buckets; one bucket per sport. For every pair of images belonging to different sports, we generate an initial NO answer with a 1.0 probability of being correct.

For any pair of images inside a bucket we ask 10 questions in AMT, without setting any specific requirements for the AMT workers that answer the questions. Then, we generate an initial answer for each image pair inside a bucket, using the 10 answers from AMT. For example, if for a specific pair we get 6 NOs out of the 10 answers, we generate an initial NO answer with probability of being correct equal to \( \frac{6}{10} \). For pairs with zero YES answers or zero NO answers, we generate an initial NO or YES answer, respectively, with a probability of being correct, 0.95. This information constitutes our initial evidence.

In order to answer the questions issued by each question selection algorithm, we get another 10 answers for each image pair inside a bucket. However, this time, we request answers from workers—experts—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. If an approach issues a 10-th question for an image pair, we return the first of the answers we retrieved for this pair. In addition, we use a smaller set of 30 images, with 5 answers per image pair, in order to estimate a human accuracy, \( p^H \), of 0.9, i.e., 9 out of 10 answers are correct, on average.

![Figure 9: AllSports: F1 score.](image)

Fig. 9 gives the F1 score, as different approaches issue new questions, until we reach a limit of 7000 questions. We use an exponential scale on the y-axis. Each curve shows the performance under a question selection algorithm and an ER algorithm. For example, in the case of bDENSE+SCC, bDENSE selects the questions and SCC performs the resolution. SCC gives an F1 score close to 0.7 using the initial evidence. Therefore, all curves under SCC start from 0.7.

After 1000 questions, bDENSE starts building a significant margin from the other SCC curves and reaches an F1 of 0.9 after only 2500 questions. On the contrary, HALFDENSE+SCC needs twice as many questions to reach an F1 of 0.9, while MLF9999+SCC stops improving its accuracy after 2000 questions and remains at an F1 of 0.8, after 6000 questions.

MLF99 asks questions on a single image pair until a YES/NO answer with a probability of at least 0.99 can be inferred. However, for a “difficult” pair of images, where we often
receive a number of consecutive false answers, MLF99 can infer a wrong answer. By applying the transitive relation on wrongly inferred image pairs, MLF99 assumes that it has fully resolved the whole dataset after 6000 questions, while the F1 accuracy under SCC is just 0.8 and under TC only 0.65. Note that MLF99+TC starts from an F1 of 0.9, because in the beginning we don’t have any expert answers, so there is no edge with a YES answer above the threshold of 0.99.

MLF9999 uses a higher probability threshold of 0.9999, to infer a YES/NO answer for each image pair. Hence, MLF9999 asks more questions for each image pair and, in exchange, it achieves a higher F1 accuracy than MLF99 eventually, under TC. Still, bDENSE gives a significantly better accuracy than MLF9999+TC, especially when a small number of questions is asked. For example, after 2500 questions bDENSE has reached an F1 of 0.9, while MLF9999+TC just reaches 0.7; which is the F1 SCC methods give using only the initial evidence. In order to reach at 0.9, MLF9999+TC needs 60% more questions than bDENSE.

6.1.2 Gymnastics

The Gymnastics dataset contains 94 images of gymnastics athletes. The main difference with the AllSports dataset is that among the 94 images there are some ones, where it is difficult to distinguish the face of the athlete, e.g. the athlete is upside down in the uneven bars. A second difference is that, in Gymnastics, more than half of the entities have 9 or more records, while, in AllSports the vast majority of entities have 3 records. In Figs. 10(a) and 10(b) we plot the distribution of records per entity in the two datasets.

![Number of Entities with a Specific Number of Records Per Entity](image1.png)

(a) AllSports  
(b) Gymnastics

Figure 10: Number of entities with a specific number of records per entity

We follow the same procedure to construct the initial answers with the one described for AllSports, using low-quality answers from AMT. The subsequent answers to the questions issued by the question selection algorithms, are provided by the high-quality pool of answers. Therefore, for each of the 4371 image pairs, we ask 5 questions to AMT workers without qualifications and 5 questions to workers with the two qualifications mentioned in Section 6.1.1.

![Images: F1 Score](image2.png)

Figure 11: Images: F1 score.

The performance of each approach for the Gymnastics dataset is depicted in Fig. 11. SCC methods clearly outperform TC methods, with bDENSE+SCC reaching an F1 just below 0.95 after 4000 questions, while MLF99+TC stays just above 0.6 and MLF9999+TC below 0.8. While MLF99+SCC stops improving after 600 questions and stays below 0.9, bDENSE+SCC keeps increasing its accuracy, even slowly, until the limit of 4000 questions.

For MLF99, we experience a behavior similar to the behavior in the AllSports dataset. MLF99 applies the transitive relation on wrongly resolved image pairs and, hence, assumes that it has fully resolved the dataset after just 600 questions. As opposed to AllSports dataset, the higher probability threshold (for inferring a YES/NO answer on an image pair) does not help MLF9999 to overcome the problem caused by “difficult” image pairs.

![Images: FPs and FNs](image3.png)

Figure 12: Images: FPs and FNs.

MLF9999+TC stays at a low F1 accuracy because of the large number of “difficult” image pairs that, for example, include a gymnast that is upside down or that is grimacing. For many of those image pairs the majority of the answers we get are wrong. Specifically, there are about 450 image pairs with a majority of wrong answers from workers with qualifications. In Fig. 12, we plot these False Negatives and False Positives, for the “expert” answers, along with the number of false answers. For instance, as we see in Fig. 12, there are 50 False Positives having 4 out of the 5 “expert” answers wrong. On the contrary, in the AllSports datasets, there are only 8 image pairs with a majority of wrong answers.

6.1.3 Cora

We use the real dataset Cora, and we generate initial answers using the Jaro similarity function. More precisely, we use the title and author fields from each record, and for a pair of records we compute the average Jaro similarity of the title and author. We then use a training set of 300, arbitrarily picked, records, to generate a mapping from similarity to probability of being the same entity, just as in [24]. For pairs of records with a probability of being the same entity, P, less than 0.5, we generate a NO answer with probability of being correct 1 − P. Then, we arbitrarily pick 500 records as our testing set. In Fig. 13(a) we plot the number of entities with a specific number of records per entity, for the 500 records of our testing set. Every time each approach selects a question to ask next, we generate a human answer synthetically, using an HA of 0.9.

![Cora Dataset](image4.png)

Figure 13: Cora dataset.
Fig. [13(b)] depicts the F1 accuracy for the 4 approaches, until each approach asks a total of 5000 questions. We see a trade-off appearing, between a) getting a high F1 score with a few questions but not “fully resolving” the dataset after 5000 questions, in bDENSE+SCC and HALF99+SCC, and b) starting “slowly” but reaching eventually an F1 of 1.0, in MLF99+TC. In particular, bDENSE+SCC and HALF99+SCC start from an F1 of 0.8 and reach 0.85 after 1000 questions. At this point, MLF99+TC gets ahead from the SCC methods and, then, after 1500 questions achieves a full resolution with an F1 of 1.0. On the other hand, bDENSE+SCC and HALF99+SCC converge to an F1 close to 0.9 after 5000 questions.

Let us now analyze the reason why bDENSE+SCC does not converge to an F1 of 1.0. We will refer to two specific entities from the dataset, in order to describe the case. Entity tagged as fahlman1990a has 10 records in our testing set, while entity tagged as fahlman1990b has 42 records. The records from both entities have the same title and authors; with one entity being the technical report, and the other entity being the corresponding conference paper. Therefore, each of the 10 records of fahlman1990a has a “strong” initial YES answer with each of the 42 records of fahlman1990b. For this reason, bDENSE can not detect a “lack”-of-evidence between the records of fahlman1990a and fahlman1990b, and does not issue questions between the two record sets; questions required for the resolution algorithm to infer that these records refer to different entities. The same case with the one taking place between the records of fahlman1990a and fahlman1990b, happens frequently in the dataset, when we use only the title and author fields, causing the low F1 for bDENSE+SCC.

In the next section, we attempt to provide deeper insights on the behavior of the five approaches under different scenarios, using synthetically generated datasets.

6.2 Synthetic Data

Besides the 2 datasets of images discussed in the previous section, we also experimented with a number of other real-world datasets. While bDENSE clearly outperforms other approaches in many cases, there are also cases where there is no significant gain or bDENSE performs worse than MLF or HALF. Instead of discussing the rest of the real-world datasets, we believe it is more instructive to study the trade-offs in a controlled environment using synthetic datasets. Such datasets let us examine how each approach is affected as we vary the main parameters involved in the dataset generation. We start by describing the dataset generation process and the main synthetic data parameters and, then, we discuss our findings for different scenarios.

The gold standard clustering along with the initial answers are generated using a synthetic data generation algorithm. The formal description of the algorithm is included in the Appendix. Here, we discuss the outline of the algorithm, focusing on the parameters controlling data generation. The algorithm receives as input the number of records the generated dataset should contain, REC$S$, the distribution controlling the number of records per entity, DISTRO, and two other parameters, BUCKET and FPR, controlling the False Positive answers, i.e., the wrong YES answers between records of different entities. The algorithm’s output is the gold standard clustering, i.e., the mapping of records to entities, and initial YES/NO answers for pairs of records.

For each initial answer, along with the YES/NO label we attach the probability of the answer being correct; the probability that generated the label.

First, the algorithm generates the gold standard clustering. The distribution of number of records per entity in the gold clustering, is given by DISTRO. Then we group entities into buckets. BUCKET entities are placed inside each bucket. Entities inside a bucket can have records that look similar. In real datasets, the records of one entity, may look similar to the records of other entities, however, the number of such entities is usually limited. The BUCKET parameter quantifies this limit and controls the distribution of initial False Positive answers. For example, if BUCKET is 10, it means that the records of one entity may be similar to records from 9 other entities; and are clearly dissimilar to the records of all other entities. Between two records from different buckets, we produce a NO answer with a very high probability, $1 - \epsilon$. In our experiments, we used an $\epsilon$ of $10^{-5}$.

Inside a bucket, we control the percentage of False Positives, i.e., the YES answers for pairs of records referring to different entities, through the FPR parameter. In particular, FPR gives the percentage of record pairs that refer to different entities, however, they are not clearly dissimilar. For such pairs we first uniformly pick a probability $p$ in $[0.5, 1.0]$. Then, we generate a NO answer with probability $p$. In the same way, we generate answers for pairs of records referring to the same entity, i.e., we first uniformly pick a probability $p$ and then create a YES answer with probability $p$. For the $1 - \text{FPR}$ percentage of record pairs that refer to different entities and are clearly dissimilar, we produce a NO answer with a very high probability, $1 - \epsilon$.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Values</th>
<th>Base Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC$S$</td>
<td>Total Dataset Records</td>
<td>(500)</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>(50, NO)</td>
<td>20</td>
</tr>
<tr>
<td>FPR</td>
<td>Controls Machine False Positives</td>
<td>(0.1, 0.4)</td>
<td>0.1</td>
</tr>
<tr>
<td>HA</td>
<td>Human Accuracy</td>
<td>(0.7, 0.9)</td>
<td>0.7</td>
</tr>
<tr>
<td>ERASE</td>
<td>Controls False Negatives</td>
<td>(0.0, 0.2)</td>
<td>0.0</td>
</tr>
</tbody>
</table>

The different question selection algorithms use the initial answers produced by the synthetic data generation process, and issue new questions to be answered by the crowd. In order to generate a human answer for a new question, we use a Human-Accuracy($HA$) parameter. A correct answer is produced with probability $HA$. However, in real datasets there may be pairs of records that refer to the same entity, still, it is very difficult for humans to infer that. As we saw in Section 5.1.2 this is often the case with the Gymnastics dataset, where for a significant number of image pairs showing the same person, the vast majority of humans give a NO answer. For this reason, we introduce an ERASE factor to reflect this behavior in our synthetic data experiments. For each entity, and out of all the pairs of the entity’s records, we “erase” a percentage of ERASE pairs. For an “erased” pair of records, the correct answer becomes NO for the initial machine answers and the subsequent human answers. For example, if $HA$ is 0.8, for an “erased” pair of records, we produce a NO answer with probability of 0.8 and a YES answer with probability 0.2. Note that it is possible to “erase”
so many edges inside an entity, that the records of the entity become “disconnected”, i.e., they can be split into two sets such that all the edges between the two sets are “erased”. Still, this occurs very rarely in the single setting we examine that uses a non-zero ERASE factor (see the discussion for Fig. 19(a)). Table 1 summarizes all the parameters used in the synthetic data experiments.

In order to generate each curve in the plots of the Synthetic Data 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.

![Figure 14: High quality initial answers (FPR= 0.1)](image)

For the base case in Table 1, Fig. 14(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 entities (DISTRO), with a mean of 3.0 and a variance of 2.0. SCC gives an F1 score close to 0.85 using the initial evidence. Therefore, all curves under SCC start from 0.85.

bDENSE significantly outperforms the other approaches by reaching an F1 of 0.95 after 2500 questions. On the other hand, MLF99+SCC needs 7500 more questions to reach an F1 of 0.95, while HALF99+SCC after 10000 questions is still below 0.95.

MLF99+TC and MLF9999+TC perform very poorly since they were not designed for a scenario with “faulty” workers. “Faulty” workers with a low accuracy of 0.7 force MLF99 and MLF9999 to ask a lot of questions for each pair of records, until the probability threshold of 0.99 or 0.9999, respectively, is reached. As a result, MLF9999+TC remains below 0.5 after 10000 questions while MLF99+TC just reaches 0.8, an F1 score lower than the starting point of SCC methods.

![Figure 15: Simulator screenshots comparing MLF and bDENSE question selection.](image)

In order to give some intuition for the gain of bDENSE compared to MLF, we discuss a toy example, visualized by our simulator in Fig. 15. In this example, the gold standard consists of one entity of 4 records, one entity of 3 records, two entities of 2 records, and one entity of a single record. In the left side of the figure, we see the initial answers, used as input by all approaches, along with the gold 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 in the right side, we see how bDENSE spends the 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. 14(b) we examine a scenario with more “reliable” workers. Thus, we keep the same parameters with the base case, however, we change human accuracy, HA, from 0.7 to 0.9. All 4 approaches converge much faster compared to the base case of Fig. 14(a). For example, bDENSE+SCC needs 75% fewer questions to reach 0.95, while MLF99+SCC needs 80% fewer questions for 0.95. Still, 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.

Note that MLF99 stops asking new questions at around 4000 questions. Nevertheless, with a probability threshold of 0.99 for the full resolution of each pair, MLF99 wrongly resolves 1 out of 100 pairs, on average. The wrong resolutions will also cause a wrong inference for a number of other pairs, through transitive relations, and prevent MLF99+TC from reaching an F1 of 1.0. On the contrary, MLF9999+TC that uses a higher threshold, reaches to an F1 of 1.0, still, it needs six times the questions bDENSE requires, to reach there.

Next, we examine the effect of lowering the quality of the initial answers. Hence, we increase the false positive rate, FPR, from 0.1 to 0.4. We start from the “faulty” workers case where human accuracy, HA, is 0.7. The rest of the parameters remain the same as in the base case. Fig. 16(a) depicts the results. With the lower-quality initial evidence SCC drops to an F1 of 0.65 when using just the initial answers, which is significantly lower than the F1 of 0.85 in the base case. bDENSE+SCC improves the initial F1 by almost 50% after 10000 questions, while MLF99+SCC improves it by 30% and HALF99+SCC only by 10%. The low human accuracy along with low quality of the initial answers, drive MLF99+TC to a very low F1 score of 0.6 after 10000 questions.

![Figure 16: Low quality initial answers (FPR= 0.4)](image)

In Fig. 16(b) we stay on the low-quality initial evidence scenario, however, we switch to the more “reliable” workers case by increasing the human accuracy, HA, from 0.7 to 0.9. As expected all approaches converge faster compared to the case of Fig. 16(a). Again, bDENSE+SCC outperforms
the other approaches by a significant margin, requiring 40% less questions to reach 0.9, compared to the second best, MLF99+SCC. Because of the low quality initial answers, MLF99 stops asking questions after around 6000 questions, compared to the 4000 questions in the case of Fig. 14(b) where FPR is 0.1. However, here, MLF99+TC just reaches 0.9 after 6000 questions, compared to the F1 of 0.95 after 4000 questions in the case of Fig. 14(b) where we have a higher quality for the initial answers.

The next direction we explore is about the structure of the gold standard clustering. We still use a GAUSSIAN DISTRO with a variance of 2.0, however, we switch the mean from 3.0 to 6.0. Therefore, in the gold clustering, each entity has twice as many records on average, compared to the base case. The rest of the parameters remain as in the base case.

In Fig. 17(a) we see that SCC methods start from a higher F1 score, just above 0.9, compared to the F1 of 0.85 in the base case. The more records for an entity, the “stronger” the connectivity inside an entity based on the initial evidence. SCC utilizes the initial evidence to detect such strong connectivity and starts with a high F1, over 0.9. The gain is even higher when SCC is combined with bDENSE for the question selection. bDENSE+SCC needs 80% fewer questions than MLF99+SCC, to reach an F1 of 0.95, and 90% fewer questions than HALF99+SCC.

In many real-world datasets the records per entity are power-law distributed. In the next experiment we switch GAUSSIAN to a POWERLAW DISTRO with an exponent of −3.0. The pdf of DISTRO is given by \( x^{-3} \), for \( x \in [1, 100] \), i.e., the number of records for an entity can not be less than 1 or more than 100. We keep the rest of the parameters as in the base case. As Fig. 17(b) shows, the behavior of the four approaches is very similar to the GAUSSIAN-with-mean-6.0 case depicted by Fig. 17(a). The main difference is that MLF99+TC achieves an F1 of almost 0.7 after 5000 questions, compared to Fig. 17(a) where MLF99+TC achieves an F1 of 0.6 after 5000 questions.

In Fig. 18 we keep the POWERLAW DISTRO with the exponent of −3.0, but we switch to the scenario with the more “reliable” workers, so we increase the human accuracy, HA, to 0.9. All approaches converge very fast to an F1 of 1.0. bDENSE+SCC gives the fastest convergence, reaching 1.0 with half of the questions required by MLF99+TC, and 25% of the questions required by HALF99+SCC.

bDENSE+SCC does not always outperform the other approaches. In particular, certain types of erroneous evidence can mislead bDENSE into asking less desirable questions. On the other hand, an approach like MLF+TC is more “conservative”. MLF+TC uses the initial evidence as hints that guide it in the question selection, but does not take such evidence into account when resolution is performed. In resolution only the edges that were “checked and double-checked” (by repeatedly asking a question over the same edge), are taken into account. Thus, even though MLF+TC may be initially slower in reaching a good clustering, in high error rate cases, it may eventually reach more accurate results. Still, reaching a more accurate result eventually, is not always the case for MLF+TC. The problem with this approach is that it can be entirely misled by “double-checked” edges that were, however, wrongly resolved, as we saw in the Gymnastics dataset. Next, we discuss one synthetic-data case where MLF+TC eventually outperforms bDENSE+SCC and one where MLF+TC is being entirely misled by wrongly resolved pairs.

Let us first describe the setting where MLF+TC eventually outperforms bDENSE+SCC. We “erase” 20% of the edges between records of an entity, i.e., for the “erased” pairs of records that refer to the same entity, the majority of answers we get from humans (and machines) is NO instead of YES. Thus, we set ERASE to 0.2. In addition, we use a POWERLAW DISTRO with an exponent of −3.0 and a human accuracy, HA, of 0.9. The rest of the parameters remain the same as in the base case.

Figure 19(a) shows the performance of all four approaches drops significantly. The effect is more obvious in the case of bDENSE+SCC. Since we are using a POWERLAW DISTRO, some entities drawn from the long tail will have a large number of records. For such an entity, subsets of the entity’s records will be “weakly” connected to the rest of the entity’s records, due to the “erased” edges. For these cases, HALF, which does not take into account the graph structure, chooses to issue questions between the weakly connected subsets of records and the rest of the records of an entity. On the contrary, bDENSE, which takes into account the graph structure, does not detect a “lack”-of-evidence between the weakly connected sets of records and the rest of the records of an entity. The large number of initial NO answers on the “erased” edges, drive bDENSE to the conclusion that there is sufficient evidence, while SCC actually needs more evidence on this part of the graph.
The case of Fig. 19(a) shows a performance similar to the one in Cora. The main difference between the synthetic data case described here and the case of Cora, is that the problem in Cora is caused by the initial False Positive answers, i.e., wrong YESes, compared to the case of Fig. 19(a) where the problem is caused by the initial False Negative answers, i.e., wrong NOs.

Our finding using the setting of Fig. 19(a) and the Cora dataset, suggests that an extension of our model that would use prior knowledge about the distribution of records-per-entity in the dataset, is an interesting direction. We discuss further such an extension in Section 8. Moreover, these two cases reveal the sensitivity of our approach to the quality of the function generating the probability of the initial answers. It appears that a “low” quality function affects significantly the performance of our approach.

Next, we explore a second case with “noisy” answers that cause MLF+TC to be misled by wrongly resolved pairs of records. We use a) a POWERLAW DISTRo with a heavier tail, i.e., we have more entities with a large number of records, and b) we do not apply bucketing for the initial answers, i.e., we may have False Positives between the records of any two entities. Therefore, we set the exponent of the POWERLAW DISTRo to $-2.0$ and BUCKET to NO, i.e., all entities are grouped into a single bucket during the initial answer generation. We also keep a high human accuracy, $HA$, equal to 0.9 and we keep the rest of the parameters the same as in the base case. Fig. 19(b) depicts the performance of the four approaches. The SCC methods start from a very high F1 and remain close to an F1 of 1.0 after 5000 questions. On the other hand, MLF99+TC reaches an F1 higher than 0.8, after 500 questions, then drops to an F1 of 0.7, and remains there after 5000 questions. MLF9999 uses a higher threshold and reaches an F1 of 0.9 after 1000 questions without, however, further improving this accuracy after 3000 questions and staying at 0.95 after 5000 questions.

The problem for MLF99+TC is the large number of False Positives in the initial answers generated. MLF99 incorrectly resolves 1 out of 100 initial False Positives, on average. The increased number of entities with a large number of records, causes severe implications when applying the transitive relation on wrongly resolved record pairs. MLF9999 needs more questions to resolve each image pair, but manages to reach a higher accuracy eventually. The results in the setting of Fig. 19(b) are similar to the ones discussed in the AllSports dataset for MLF99 and MLF9999.

7. RELATED WORK

Entity Resolution is a well-studied problem and goes by various names including record linkage, deduplication, identity resolution, object identification, merge/purge, reference reconciliation. Surveys [9] and [21] discuss approaches and algorithms proposed for Entity Resolution.

Recently, a number of approaches have been proposed for enhancing ER with evidence provided by humans [10, 11, 17, 18, 20]. Reference [17] 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 [17] 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 [10]. The main focus of [10] is the probabilistic aggregation of human tasks’ outcome into an overall clustering.

As opposed to [10] and [17], references [11, 18, 20] use a simpler interface for the human task, involving only two records. Both [18] and [20] 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 [11], however, the main focus of [11] is designing an Entity Resolution Algorithm (ERA) that can handle human errors. The question selection process of [11] 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.

References [8, 14, 19] study similar problems to the question selection for ER. Reference [8] 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 [14] and [19] focus on spectral clustering [15], 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 [14] and [19] focus in the 2-way partitioning case. As opposed to [8, 14, 19], 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 [7]. In fact, the ML clustering, presented in Section 2.2, can be reduced to the weighted version of correlation clustering, as [6] discusses.

8. FUTURE WORK

There are a number of possible extensions on the problem setting that we have not considered in this paper. Here, we describe two of them and, in addition, we discuss three more topics for future work. We believe that these five directions constitute the most promising ground to further improve the efficiency of our approach.

In the first extension, we can take into account prior knowledge about the distribution of records referring to the same entity, e.g., power-law or gaussian. As we discussed in our experimental evaluation, there are cases where our approach could greatly benefit from such knowledge. As a second extension, we can consider a different probability of a human making an error for each pair of records. That is, we can estimate the error rate on a specific pair of records based on the answers we get back for this pair, e.g., if the YES/NO answers are balanced, the error rate must be “high” for that pair.

In the experimental evaluation, we mentioned that our approach is sensitive on the initial evidence, in some cases. The thorough study for the effect of the initial answers’ quality to our approach’s effectiveness is an interesting direction for future work. Another direction is the study of graph algorithms that try to find the pair of subgraphs with the highest $\rho$-ratio between them. In this paper, we proposed a simple heuristic for finding disjoint sets with a high $\rho$-ratio, however, more sophisticated algorithms can give a better outcome and, potentially, improve the overall accuracy of our approach. Finally, it would be interesting to explore the effect of other ERA, when applied using the evidence that bDENSE collects. For example, an ERA with performance
closer to the ML clustering could yield even better results than SCC.

9. 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).

10. REFERENCES

[1] AllSports. stanford.edu/~verroios/datasets/AllSports.zip
[3] Cora dataset. people.cs.umass.edu/~mccallum/data/cora-refs.tar.gz

Appendix

Algorithm 3 Synthetic Dataset Generation

Input: RECs: The number of records in the dataset
DISTRO: The distribution controlling the number of records per entity
BUCKET: How many entities to place in each bucket
FPR: The False-Positive-Rate inside a bucket

Output: Gold standard clustering and one initial answer for each pair of records

1: {Entities Generation}
2: while the number of records generated are less than RECs do
3: Using DISTRO generate a number n
4: Create a new entity with n records
5: end while
6: {Buckets Generation}
7: while there one or more entities left do
8: Randomly pick BUCKET entities
9: Group the selected entities’ records into a bucket
10: Remove the selected entities from the set of all entities
11: end while
12: {Initial Answers Generation}
13: for each bucket do
14: for each pair of records inside the bucket do
15: if the two records refer to the same entity then
16: Pick uniformly a probability p in [0.5, 1.0)
17: Pick uniformly a number x in [0.5, 1.0)
18: if x < p then
19: Create a YES answer with p attached
20: else
21: Create a NO answer with p attached
22: end if
23: else
24: Pick uniformly a number x in [0.5, 1.0)
25: if x > FPR then
26: Pick uniformly a number x in [0.5, 1.0)
27: if x > ϵ then
28: Create a NO answer with 1 − ϵ attached
29: else
30: Create a YES answer with 1 − ϵ attached
31: end if
32: else
33: Pick uniformly a probability p in [0.5, 1.0)
34: Pick uniformly a number x in [0.5, 1.0)
35: if x < p then
36: Create a NO answer with p attached
37: else
38: Create a YES answer with p attached
39: end if
40: end if
41: end if
42: end for
43: end for
44: for each pair of records between buckets do
45: Pick uniformly a number x in [0.5, 1.0)
46: if x > ϵ then
47: Create a NO answer with 1 − ϵ attached
48: else
49: Create a YES answer with 1 − ϵ attached
50: end if
51: end for