Entity-centric Attribute Feedback for Interactive Knowledge Bases

Ari Kobren, Nicholas Monath, Andrew McCallum
College of Information and Computer Science
University of Massachusetts Amherst
{akobren, nmonath, mccallum}@cs.umass.edu

1 Introduction

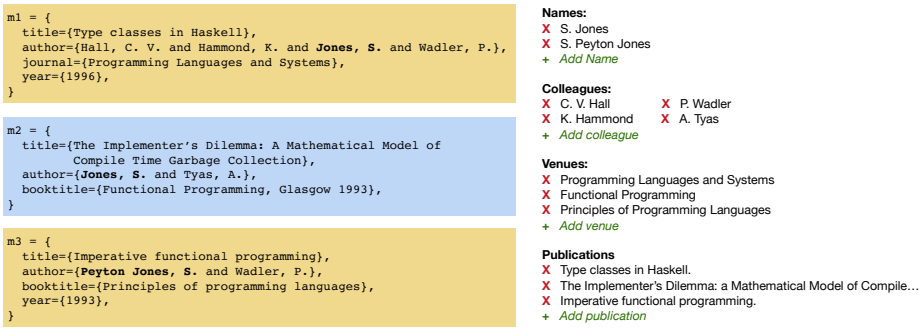
Structured data repositories, more commonly known as knowledge bases (KBs), play a central role in many important applications. For example: Google Maps[1] is based on a KB of points of interest and their attributes and Google Scholar[2] is built on top of a KB of scientists, their publications and collaborations. Since automatically constructed KBs often exhibit incorrect or missing information, leveraging humans–and particularly those who interact with the KB–to help with KB curation can be particularly effective. Some KBs are already designed to solicit and integrate user feedback: NELL prompts users to validate facts it has inferred [13]; Google Maps[3] has begun to ask users yes-or-no questions about the attributes of places they have visited; and Angeli et al. use active learning to collect labels for training a relation extraction system for KB construction [3, 4].

Effectively leveraging human workers to improve entity resolution (ER)–a prominent component of many automated KB construction approaches–is a difficult task that has recently become the subject of significant study. In both theory and practice, strategies for ER with a human-in-the-loop are centered around using crowdsourcing to collect pairwise labels (i.e., whether two mentions refer to the same ground-truth entity) [17, 15, 7, 16, 11]. While these approaches may be effective, they also have a number of disadvantages, especially in the context of KB construction. First, crowd workers are known to be noisy and have been shown to perform poorly on difficult domain specific tasks [8]. Second, these approaches do not leverage users browsing the KB who are likely to be more familiar with the KB's content than arbitrary crowd workers. It is possible to target KB users with pairwise label solicitations but labeling requests can be disruptive (without costly UI design). Moreover, pairwise techniques limit the information content of user feedback. For example, an expert may be willing to provide far richer feedback than pairwise labels if presented with the opportunity.

In this work, we develop *entity-centric attribute feedback*, a framework for user interaction and automatic feedback integration that addresses each of these issues. In our model, we assume that users interact with a KB similar to Google Scholar. Each KB entity has a corresponding profile that lists its relations and attributes (e.g., papers and collaborators). Rather than ask users to label mention pairs, in our model, while browsing a particular profile, a KB user may identify any incorrect and missing attributes. Instead of simply adding or subtracting those attributes from the profile, the feedback may trigger a repartitioning of the underlying mentions, which can resolve many ER mistakes at once [18]. Unlike methods based on pairwise labeling, our browsing-based model is non-intrusive to user experience and allows users to provide feedback where they have expertise–which mitigates noise (especially in comparison to crowd workers). Attribute feedback is also powerful: a single piece of attribute feedback can contain as much information as multiple pairwise labels.

[1] maps.google.com
[2] scholar.google.com
[3] maps.google.com

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) Mentions. (b) Profile.

Figure 1: **KB Sketch.** The attribute projection of 3 mentions–m_1, m_2, m_3 (Figure 1a)–is displayed in the profile (Figure 1b). Users may augment the profile with new attributes (green text) or reject attributes (red 'X'). m_1 and m_3 are mentions of the same entity (indicated by the yellow color).

We compare entity-centric attribute feedback to pairwise labeling approaches in simulations of interactive ER. We experiment with two real-world datasets: one containing citations records with ambiguous author names and the other containing ambiguous patent assignees. Our experiments reveal that attribute feedback is superior to pairwise labels with respect to the number of queries necessary to discover the ground-truth. In one experimental setting, we show that the same algorithm requires 70% fewer queries when it solicits attribute feedback than when it solicits pairwise labels.

2 Entity-centric Attribute Feedback

We provide an overview of KB construction via entity resolution (ER) and give an example of our interactive framework. Then, we define ER and present formal models for: mentions and entities, attribute feedback and interaction with a KB.

Interactive KB Sketch. A KB is built from a collection of mentions. Each mention refers to exactly one real-world entity. For example, Figure 1a contains three mentions of `S. Jones`, where m_1 and m_3 refer to one real-world entity (the yellow entity) and m_2 refers to a different real-world entity (the blue entity). An ER algorithm is used to partition the mentions into clusters. A profile page is built for each cluster using the attributes of the corresponding mentions (Figure 1b). Users interact with profiles by identifying incorrect and missing attributes.

Entity Resolution. More formally, the input to ER is a set of *mentions*, $\mathcal{M} = \{m_i\}_{i=0}^{n}$. Each mention, m, refers to a single ground-truth entity, $C^\star(m) = e^\star$. The output of an ER algorithm is a partition of the mentions; each clustering of the partition is known as an *inferred entity*, $\hat{C}$. The goal is to recover a partition in which each inferred entity contains mentions that all refer to the same ground-truth entity and in which no two inferred entities correspond to the same ground-truth entity. ER is performed using a learned pairwise similarity model, $S : \mathcal{M} \times \mathcal{M} \to [0, 1]$.

Data Model. Both mentions and entities possess attributes. For a mention m, if $C^\star(m) = e^\star$ then the attributes of m must be a subset of the attributes of $e^\star$, i.e, $A(m) = \{a_i, \cdots, a_j\} \subseteq A(e^\star)$, where $A(\cdot)$ returns the attributes of its argument. The *attribute projection* of an inferred entity, $\hat{C}$, is the union of the attributes of all its mentions: $P_A(\hat{C}) = \bigcup_{m:\hat{C}(m)} A(m)$. To mitigate ambiguity, the mentions and entities must obey the following two properties:

1. $\forall e_1^\star, e_2^\star \in \mathcal{E}^\star, \quad A(e_1^\star) \not\subseteq A(e_2^\star) \wedge A(e_2^\star) \not\subseteq A(e_1^\star)$; and
2. $\forall m \in \mathcal{M}, \forall e^\star \in \mathcal{E}^\star \backslash C^\star(m) \quad A(m) \not\subseteq A(e^\star)$.

Here, $\mathcal{E}^\star$ is the set of ground-truth entities.

The first property guarantees that no ground-truth entity possesses a collection of attributes that are a subset of any other ground-truth entity's attributes. The second property guarantees that each

mention's attributes are a subset of exactly one ground-truth entity's attributes. These properties are intuitive and in our experiments with real-world data (Section 4), the properties hold. Note that these two properties neither restrict two ground-truth entities nor two mentions from having some attributes in common, even if both mentions refer to different ground-truth entities.

Attribute Feedback. When browsing the profile of an inferred entity, the user sees its attribute projection (as in Figure 1b). If the user identifies an attribute, a, that she believes is incorrect, she can provide a *rejection*, f_i^{-a} (i indexes the total feedback count). For example, in Figure 1b, the user can click the red 'X' next to `A. Tyas` to express the belief that `S. Jones` never coauthored with `A. Tyas`. Alternatively, the user may *augment* the profile with a missing attribute, a', notated, $f_i^{+a'}$.

In the attribute feedback framework, two seemingly opposite pieces of feedback can both be considered correct. For example, in the profile in Figure 1b, a user can reject the coauthor `A. Tyas` if she believes that the profile belongs to the yellow entity, or, the user can reject the coauthor `P. Wadler` if she believes that the profile belongs to the blue entity. Neither piece of feedback is incorrect. However, attribute feedback can be characterized as *inconsistent* with the ground-truth. A rejection f_i^{-a} made with respect to an inferred entity, $\hat{C}$, is *inconsistent* with the ground-truth if, $\forall m \in \hat{C},\ a \in P_A(C^\star(m))$. In the example above, if a cluster contained m_1 and m_3 (but not m_2), the rejection of any attribute would be inconsistent with the ground-truth. An augmentation f_i^{+a} is inconsistent with the ground-truth if, $\nexists m \in \hat{C},\ a \in P_A(C^\star(m))$.

Inference for Attribute Feedback. Attribute feedback is a powerful style of interaction. For example, consider a rejection f_i^{-a} made with respect to $\hat{C} = \{m_i, m_j\}$. Without loss of generality, assume that $a \in A(m_j)$. If the feedback is consistent then m_i does not belong to the same cluster as *any mention that contains* a. This set contains at least one mention (m_j) but possibly many more.

Attribute feedback can also be ambiguous. Consider a rejection, f_i^{-a}, made with respect to $\hat{C} = \{m_i, m_j, m_k\}$. Without loss of generality, assume that neither m_i nor m_k contains the attribute a. Even if the feedback is consistent, the precise target(s) of the feedback are ambiguous. That is, letting $\{\cdot\}$ denote a partition and $|$ a cluster boundary, the following partitions respect the feedback: $\{m_i, m_j | m_k\}$, $\{m_i | m_j | m_k\}$, $\{m_i | m_j, m_k\}$ and $\{m_i, m_k | m_j\}$. Note that the only partition that does not respect the feedback is the complete partition, $\{m_i, m_j, m_k\}$. As a concrete example, consider the rejection of `A. Tyas` from the profile above; the only partition of m_1, m_2, m_3 that does not respect such a rejection is the complete partition.

Entity-centric Interaction Model. Our interaction model proceeds in rounds. In each round the user selects an inferred entity, $\hat{C}$; this simulates the user browsing the corresponding profile in the KB. The user is shown $P_A(\hat{C})$, i.e., the attribute projection of $\hat{C}$ (as in Figure 1b). The user may then provide attribute feedback with respect to $\hat{C}$. The KB stores the feedback, ER is rerun and the next round begins. Note that this differs from virtually all interaction models for ER in two ways. First, the interaction style is entity-centric, i.e., the KB can only receive feedback made with respect to $\hat{C}$, an inferred entity of the user's choosing. Second, the feedback is made with respect to the attributes of $\hat{C}$ and not directly with respect to any (pair of) mentions.

3 Simulation Framework

We design a framework for simulating interactive ER so that we may compare entity-centric attribute feedback and pairwise feedback. Our framework is inspired by ER with an oracle [17, 15, 7, 5, 10, 1]. While ER with an oracle is somewhat unrealistic, it provides a formal basis for analyzing and experimenting with various types of feedback. In the following descriptions, the mentions form a graph with an edge between each pair. If a pair of mentions refer to the same ground-truth entity the corresponding edge is positive; all other edges are negative (see Figure 2).

ER with an Oracle (ERO). This framework proceeds in rounds. During each round, an ER algorithm adaptively queries the oracle with a pair of mentions. The oracle's feedback is either positive or negative indicating that the pair either do or do not refer to the same ground-truth entity, respectively. Upon receiving a positive or negative label, the signs of additional edges may be inferred

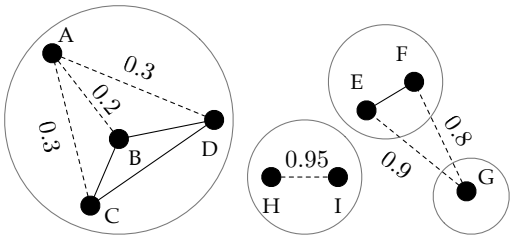


Figure 2: Black circles (labeled A–I) and gray circles represent mentions and inferred entities, respectively. Solid black lines and missing lines between mentions indicate known positive and negative edges, respectively. Dashed lines indicate unknown edges with similarities annotated.

via transitivity. Interaction ends when the sign of each edge is known (or inferrable). The goal is two-fold: first, minimize the number of queries to the oracle and second, maximize recall (i.e., known positive edges) as efficiently as possible.

KB ER with an Oracle (K-ERO). Rather than allow the ER algorithm to select any edge (i.e., mention pair), each round begins with the oracle selecting an inferred entity, $\hat{C}$. Next, the ER algorithm chooses a pair of mentions such that at least one of the mentions belongs to $\hat{C}$–thereby satisfying entity-centricty (Section 2). The oracle labels the edge as before. Unlike ERO, after each round of interaction, the ER algorithm constructs a (potentially) new partition of the mentions. As before, a primary goal is to minimize the number of queries to the oracle. However, in K-ERO, rather than recall, a secondary goal is to maximize the pairwise F1-score of the partition of the mentions in each round. This is appropriate in that F1-score is a common measurement for evaluating ER [9, 12].

ERO with Attribute Feedback. Both ERO and K-ERO can be adapted for attribute feedback. Instead of querying the oracle with a mention pair, in the attribute feedback setting, the oracle is shown the corresponding attribute projection. The oracle may provide a consistent rejection or augmentation (Section 2). For some pairs of mentions, neither a consistent rejection nor a consistent augmentation can be returned; in these cases the oracle my respond with "all good."

Attribute feedback has two advantages over pairwise feedback. First, when the oracle rejects attribute a with respect to a queried pair, negative edges are inferrable between one of the queried mentions and all mentions that exhibit the attribute a. This is at least as powerful as a negatively labeled edge in the pairwise setting. When the oracle augments a projection or responds with "all good," the corresponding edge is positive. But, an augmentation also provides additional information about a missing attribute; this information can be utilized by inference. Consider querying the oracle with the edge $\{m_i, m_j\}$ and a resulting augmentation, f_i^{+a}. If, after labeling the edge positively, the corresponding inferred entity does not contain the attribute a, ER may make additional merges for the sake of respecting the augmentation. This can help improve the F1-score.

3.1 Query Strategies

We test four strategies for selecting edges with which to query the oracle. All methods have access to a similarity function, $S : \mathcal{M} \times \mathcal{M} \to [0, 1]$ (in our work, and in other related work, the similarity of an edge is treated as the probability that the edge is positive). For each method, we also experiment with a corresponding entity-centric version which must select an edge such that at least one endpoint is in the inferred entity chosen by the oracle (thereby obeying entity-centricity). See Figure 2 for a visual illustration of a partition of the mentions and the corresponding edge selected by each strategy.

WANG [17]. In each round, this strategy selects the unknown edge with the largest similarity score to be labeled by the oracle. In Figure 2, WANG selects edge (H, I) because it has the highest score.

WANG-F1. This strategy is inspired by the WANG strategy but is designed to optimize for F1-score rather than recall. In each round, WANG-F1 finds the most similar across-cluster edge and the least similar within-cluster edge. In terms of F1-score, labeling an across-cluster edge as positive leads to a merger and thus an increased number of true positives; labeling a within-cluster edge as negative splits the cluster reducing the number of false positives. Let $e_0 = (i^+, j^+)$ and $e_1 = (i^-, j^-)$ be the most

and least similar such edges, respectively. Then, WANG-F1 selects $\text{argmax}_{e \in \{e_0, e_1\}} |S(e) - 0.5|$, i.e., the edge most likely to induce a split or merge. In Figure 2, WANG-F1 selects edge (E, G)—the unknown, cross-cluster edge with the maximum value of $|S(e) - 0.5|$.

EDGE [7]. This strategy selects the unknown edge with the largest *benefit*. The benefit of an unknown edge is $b_e(u, v) = S(u, v) \cdot |c_T(u)| \cdot |c_T(v)|$ where $c_T(u)$ is the number of mentions that have positive edges to u. In Figure 2, EDGE selects (E,G) because it has the largest benefit ($b_e(E, G) = 0.9 \cdot 2 \cdot 1$).

EDGE-F1. For each edge (u, v), EDGE-F1 computes the benefit of that edge, $b_e(u, v)$. If (u, v) is a within-cluster edge, EDGE-F1 also computes the expected number of false positives reduced if that edge is labeled negatively: $r_e(u, v) = (1 - S(u, v)) \cdot |c_T(u)| \cdot |c_T(v)|$. If (u, v) is an across-cluster edge then $r_e(u, v) = 0$. EDGE-F1 selects the edge with highest value of $b_e(\cdot, \cdot) + r_e(\cdot, \cdot)$. In this way, EDGE-F1 tries to select an edge such that after labeling the edge and repartitioning the mentions, the F1-score will be most improved. In Figure 2, EDGE-F1 selects edge (A, B) because it has the largest expected effect on the F1-score ($b_e(A, B) + r_e(A, B) = 0.2 \cdot 1 \cdot 3 + 0.8 \cdot 1 \cdot 3$).

4 Experiments

We compare pairwise (P) and attribute (A) feedback in the unconstrained (ERO) and entity-centric (K-ERO) settings (Section 3). For each feedback style and setting, we experiment with 4 query strategies (Section 3). We measure the F1-score per query and total number of queries required for each combination of: feedback style, interaction setting and query strategy. We run experiments on 2 real-world datasets: the REXA author name coreference dataset [6] and a subset of the assignee records from the PatentsView[4] database with annotation by NBER[5]. In order to expedite our experiments, we limit the number of edges considered by EDGE and EDGE-F1 strategies in each round to 1000.

4.1 Setup

For the REXA dataset, we train a pairwise classifier that scores the similarity of two mentions. We use a linear model with features similar to [14, 19]. For the patents data, we set the similarity of two mentions to be the Jaro-Winkler distance between the corresponding assignees [20]. All similarity scores for Rexa are transformed to $[0, 1]$ via a sigmoid (the Jaro-Winkler scores are already in$[0, 1]$).

We use KwikCluster to create an initial partition of the mentions with respect to their similarity scores [2]. We also experiment with initialization to the complete partition (i.e., a single cluster of all mentions) and the shattered partition (i.e., each mention in a singleton cluster). The complete and shattered initializations help us to understand how the strategies perform in the case of severe over- and under-merging. In our experiments, these partitions are used as the starting KB state.

4.2 Results

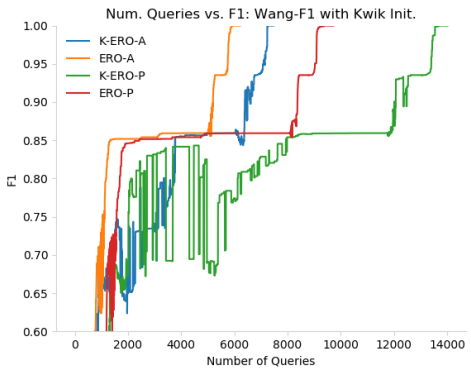
Figure 4 shows the number of queries required to discover the ground-truth for both datasets under various settings. In the entity-centric setting (K-ERO), attribute feedback leads to faster discovery of the ground-truth partition than pairwise feedback in 21 out of 24 experimental conditions. In the unconstrained setting (ERO), attribute feedback is always better than pairwise feedback. In some cases, the advantage due to attribute feedback is very significant: WANG-F1–one of the strongest strategies across datasets and initializations–requires 70% fewer queries when receiving entity-centric attribute feedback than when receiving entity-centric pairwise feedback for the PatentsView dataset.

None of the query strategies is always best. However, the results show that the EDGE and WANG-F1 strategies are often most performant. The EDGE-F1 strategy is often less performant than the other methods. This is most pronounced by the fact that it performs worse in the unconstrained pairwise setting than the entity-centric setting. EDGE-F1 could be hindered by its preference for querying edges that are likely to be negative. This suggests that rapid discovery of positive edges is important for the sake of both minimizing the number of queries to the oracle and maximizing F1-score.

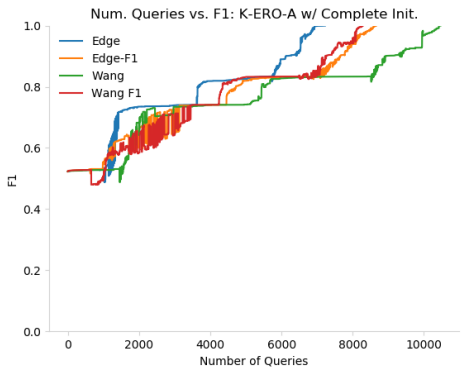
Finally, we investigate the improvement in F1-score, per query, of each strategy. Across settings, we find that no strategy consistently improves the F1-score fastest (see Figure 3b for an example).

[4] `http://www.patentsview.org/`

[5] `https://sites.google.com/site/patentdataproject/Home/downloads`



(a) Effect of attribute feedback (Rexa).



(b) Impact of query strategy (Rexa).

Figure 3: Number of queries vs. micro F1 (across all blocks/canopies) with respect to different query strategies and feedback styles.

Set.	W-K	W-C	W-S	E-K	E-C	E-S	W-F-K	W-F-C	W-F-S	E-F-K	E-F-C	E-F-S
K-A	13382	**10466**	9683	**7477**	**7224**	**6655**	**7480**	**8288**	**6451**	**8094**	**8657**	**6502**
K-P	**12957**	10506	**9672**	15724	11284	9959	14014	15171	9696	23314	15918	9908
A	**6204**	**6135**	**6291**	**6563**	**6616**	**6573**	**6221**	**6322**	**6097**	**10673**	**10890**	**10790**
P	9500	9500	9500	10109	10109	10109	9727	19234	9500	32523	32550	32521

(a) Rexa: number of queries.

Set.	W-K	W-C	W-S	E-K	E-C	E-S	W-F-K	W-F-C	W-F-S	E-F-K	E-F-C	E-F-S
K-A	**35890**	**21785**	21904	**12682**	**9473**	**9453**	**9340**	**8739**	**7470**	**12469**	**9653**	**9599**
K-P	37041	21792	**21888**	47497	25568	25677	31655	69871	21860	47120	25441	26090
A	**7353**	**7350**	**7444**	**8546**	**8549**	**8608**	**10285**	**8382**	**7780**	**8562**	**8479**	**8593**
P	21735	21735	21735	23096	23096	23096	21735	64157	21735	23093	23094	23093

(b) PatentsView: number of queries.

Figure 4: Number of queries for Rexa and PatentsView datasets. K-A, K-P, A, and P represent the attribute and pairwise variants of the K-ERO and ERO settings, respectively. W, E, W-F and E-F represent WANG, EDGE, WANG-F1 and EDGE-F1 strategies, respectively. K, C and S represent KWIK, COMPLETE and SHATTERED initializations.

This further highlights the importance of discovering positive edges over negative edges, even when the initialization contains significant over-merging. The plots also imply that feedback style (i.e., attribute or pairwise) has a larger effect on performance than the query strategy.

5 Discussion and Conclusion

In this work we introduce entity-centric attribute feedback, a new, formal style of KB interactivity. Unlike previous approaches based on actively soliciting arbitrary users (e.g., crowd workers) for pairwise labels, our approach makes it natural to collect feedback from users browsing the KB, especially with respect to entities that they are likely to be familiar with. We define a formal model for entities, mentions and attributes, discuss the power of attribute feedback and demonstrate its superiority to pairwise feedback in experiments on real-world data.

Our work adds an important new component to the landscape of interactive KB construction and illuminates a number of interesting areas for future investigation. For example, as in the recent work on ERO, we are interested in designing new query strategies and formally studying their query complexity in the K-ERO setting. Analyzing the K-ERO setting with a noisy oracle is also an important next step (we note that noisy oracles have been rigorously analyzed in the ERO setting [10]). We are simultaneously interested in designing scalable integration algorithms for user feedback. These algorithms will require the development of new data structures and representations of user feedback. Our work provides the tools for both theoretical and empirical study of interactive KBs. We hope that the entity-centric attribute feedback framework spurs increased discussion of methods for interactive KB curation among members of the AKBC community.

References

[1] Nir Ailon, Anup Bhattacharya, Ragesh Jaiswal, and Amit Kumar. Approximate clustering with same-cluster queries. *arXiv:1704.01862*, 2017.

[2] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. *JACM*, 2008.

[3] Gabor Angeli, Sonal Gupta, Melvin Jose, Christopher D. Manning, Christopher Ré, Julie Tibshirani, Jean Y. Wu, Sen Wu, and Ce Zhang. Stanford's 2014 slot filling systems. *TAC KBP*, 2014.

[4] Gabor Angeli, Julie Tibshirani, Jean Y. Wu, and Christopher D. Manning. Combining distant and partial supervision for relation extraction. *EMNLP*, 2014.

[5] Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. *NIPS*, 2016.

[6] Aron Culotta, Pallika Kanani, Robert Hall, Michael Wick, and Andrew McCallum. Author disambiguation using error-driven machine learning with a ranking loss function. *IIWeb*, 2007.

[7] Donatella Firmani, Barna Saha, and Divesh Srivastava. Online entity resolution using an oracle. *VLDB*, 2016.

[8] Ari Kobren, Thomas Logan, Siddarth Sampangi, and Andrew McCallum. Domain specific knowledge base construction via crowdsourcing. *AKBC*, 2014.

[9] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[10] Arya Mazumdar and Barna Saha. Clustering with noisy queries. *arXiv:1706.07510*, 2017.

[11] Arya Mazumdar and Barna Saha. Query complexity of clusting with side information. *arXiv:1706.07719*, 2017.

[12] David Menestrina, Steven Euijong Whang, and Hector Garcia-Molina. Evaluating entity resolution results. *VLDB*, 2010.

[13] Tom M Mitchell, William W Cohen, Estevam R Hruschka Jr, Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, et al. Never ending learning. *AAAI*, 2015.

[14] Pucktada Treeratpituk and C Lee Giles. Disambiguating authors in academic publications using random forests. *JCDL*, 2009.

[15] Norases Vesdapunt, Kedar Bellare, and Nilesh Dalvi. Crowdsourcing algorithms for entity resolution. *VLDB*, 2014.

[16] Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. Crowder: Crowdsourcing entity resolution. *VLDB*, 2012.

[17] Jiannan Wang, Guoliang Li, Tim Kraska, Michael J Franklin, and Jianhua Feng. Leveraging transitive relations for crowdsourced joins. 2013.

[18] Michael Wick, Ari Kobren, and Andrew McCallum. Probabilistic reasoning about human edits in information integration. 2013.

[19] Michael Wick, Sameer Singh, and Andrew McCallum. A discriminative hierarchical model for fast coreference at large scale. *ACL*, 2012.

[20] William E Winkler. The state of record linkage and current research problems. *Statistics of Income Division, Internal Revenue Service Publication R99/04*, 1999.