Collective Entity Resolution in Familial Networks

Pigi Kouki*, Jay Pujara*, Christopher Marcum†, Laura Koehly†, and Lise Getoor*
*School of Engineering, University of California Santa Cruz
Email: pkouki@soe.ucsc.edu, jay@cs.umd.edu, getoor@soe.ucsc.edu
†National Human Genome Research Institute, National Institutes of Health
Email: chris.marcum@nih.gov, koehlyl@mail.nih.gov

***Abstract*—Entity resolution in settings with rich relational structure often introduces complex dependencies between co-references. Exploiting these dependencies is challenging – it requires seamlessly combining statistical, relational, and logical dependencies. One task of particular interest is entity resolution in familial networks. In this setting, multiple partial representations of a family tree are provided, from the perspective of different family members, and the challenge is to reconstruct a family tree from these multiple, noisy, partial views. This reconstruction is crucial for applications such as understanding genetic inheritance, tracking disease contagion, and performing census surveys. Here, we design a model that incorporates statistical signals, such as name similarity, relational information, such as sibling overlap, and logical constraints, such as transitivity and bijective matching, in a collective model. We show how to integrate these features using probabilistic soft logic, a scalable probabilistic programming framework. In experiments on real-world data, our model significantly outperforms state-of-the-art classifiers that use relational features but are incapable of collective reasoning.**

I. Introduction

Entity resolution, the problem of identifying, matching, and merging references corresponding to the same entity within a dataset, is a widespread challenge in many domains. Here, we consider one particularly compelling application, the problem of entity resolution in familial networks, which is an essential component in applications such as social network analysis [13], medical studies [20], family health tracking and electronic healthcare records [14], genealogy studies [10], and areal administrative records, such as censuses [27]. Familial networks contain a rich set of relationships between entities with a well-defined structure, which differentiates this problem setting from general relational domains such as citation networks that contain a fairly restricted set of relationship types.

As a concrete example of entity resolution in familial networks, consider healthcare records from several patients from a single family. Each patient supplies a family medical history, identifying the relationship to an individual and their symptoms. One patient may report that his 15-year old son suffers from high blood sugar, while another patient from the same family may report that her 16-year old son suffers from type 1 diabetes. Assembling a complete medical history for this family requires determining whether the two patients have the same son and are married.

In this setting, a subset of family members independently provide a report of their familial relationships. This process yields several ego-centric views of *a portion* of a familial network, i.e., persons in the family together with their relationships. Our goal is to infer the entire familial network by identifying the people that are the same across these ego-centric views. For example, in Figure 1 we show two partial trees for one family. In the top tree, the patient "Jose Perez" reported his family tree and mentioned that his 15-year old son, also named "Jose Perez," has high blood sugar. In the bottom tree, the patient "Anabel Perez" reported her family tree and mentioned that her 16-year old son suffers from type 1 diabetes. In order to assemble a complete medical history for this family we need to infer which references refer to the same person (indicated by the same colors), e.g., that "Ana Maria Perez" from the top tree is the same person with "Anabel Perez" from the bottom tree.

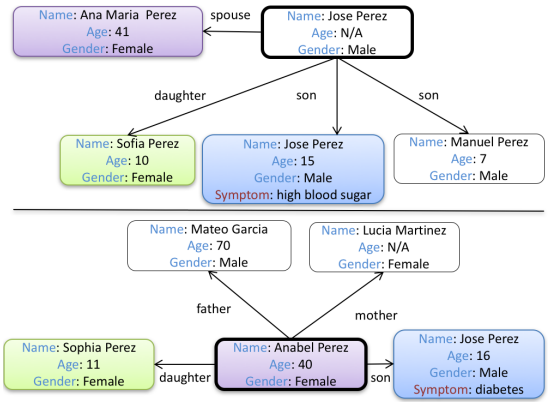

Fig. 1: Two familial ego-centric trees. Bold black borders indicate the root of the tree. Persons in same color represent same entities. White means that the persons were not matched across the trees.

Typical approaches to performing entity resolution use attributes characterizing a reference (e.g., name, occupation, age) to compute different statistical signals that capture similarity, such as string matching for names and numeric distance for age [27]. However, relying only on attribute similarity to perform entity resolution in familial networks is problematic since these networks present unique challenges: attribute data is frequently incomplete, unreliable, and/or insufficient. Participants providing accounts of their family frequently forget to include family members or incorrectly report attributes, such as ages of family members. In other cases, they refer to the names using alternate forms. For example, consider the two ego-centric trees of Figure 1. The top tree contains one individual with the name "Ana Maria Perez" (age 41) and

the bottom one an individual with the name "Anabel Perez" (age 40). In this case, using name and age similarity only, we may possibly determine that these persons are not co-referent, since their ages do not match and the names vary substantially. Furthermore, even when participants provide complete and accurate attribute information, this information may be insufficient for entity resolution in familial networks. In the same figure, the top tree contains two individuals of the name "Jose Perez", while the bottom tree contains only one individual "Jose Perez." Here, since we have a perfect match for names for these three individuals, we cannot reach a conclusion which of the two individuals of the top tree named after "Jose Perez" match the individual "Jose Perez" from the bottom tree. Additionally using age similarity would help in the decision, however, this information is missing for one person. In both cases, the performance of traditional approaches that rely on attribute similarities suffers in the setting of familial trees.

In this scenario, there is a clear benefit from exploiting *relational information* in the familial networks. Approaches incorporating relational similarities [4], [9], [16] frequently outperform those relying on attribute-based similarities alone. Recently [25], *collective* approaches where related resolution decisions are made jointly, rather than independently, showed improved entity resolution performance, albeit with the tradeoff of increased time complexity. General approaches to collective entity resolution have been proposed [23], but these are generally appropriate for one or two networks and do not handle many of the unique challenges of familial networks. Accordingly, much of the prior work in collective, relational entity resolution has incorporated only one, or a handful, of relational types, has limited entity resolution to one or two networks, or has been hampered by scalability concerns.

In contrast to previous approaches, we develop a scalable approach for collective relational entity resolution across multiple networks with multiple relationship types. Our approach is capable of using incomplete and unreliable data in concert with the rich multi-relational structure found in familial networks. We view the problem of entity resolution in familial networks as a collective classification problem and propose a model that can incorporate statistical signals, relational information, and logical constraints. Our model is able to collectively reason about entities across networks using these signals, resulting in improved accuracy. To build our model, we use *probabilistic soft logic* (PSL) [3], a probabilistic programming framework which uses soft constraints to specify a joint distribution over possible entity matchings. PSL is especially well-suited to entity resolution tasks due to its ability to unify attributes, relations, and constraints such as bijection and transitivity into a single model.

Our contributions mirror the structure of this paper. In Section II we formally define the problem of entity resolution for familial networks. Section III introduces a process of *normalization* that enables the use of relational features for entity resolution in familial networks. In Section IV, we develop a scalable entity resolution framework that effectively combines attributes, relational information, and logical constraints. Section V first presents an extensive evaluation on two real-world datasets, from real patient data from the National Institutes of Health and Wikidata, demonstrating that our approach beats state-of-the-art methods while maintaining scalability as problems grow. Next, we provide a detailed analysis of the features most useful for relational entity resolution, providing advice for practitioners. Section VII briefly surveys the related approaches to relational entity resolution. Finally, Section VII highlights several potential applications for our method and promising extensions to our approach.

II. Problem Setting

We consider the problem setting where we are provided a set of ego-centric *reports* of a familial network. Each report is given from the perspective of a *participant* and consists of two types of information: family members and relationships. The participant identifies a collection of family members and provides personal information such as name, age, and gender for each person (including herself). The participant also reports their relationships to each family member, which we categorize as first-degree relationships (mother, father, sister, daughter, etc.) or second-degree relationships (grandfather, aunt, nephew, etc.). Our task is to align family members *across* reports in order to reconstruct a complete family tree. We refer to this task as *entity resolution in familial networks* and formally define the problem as follows:

Problem Definition. We assume there is an underlying family $\mathbf{F} = \langle \mathbf{A}, \mathbf{Q} \rangle$ which contains (unobserved) actors $\mathbf{A}$ and (unobserved) relationships $\mathbf{Q}$ amongst them. We define $\mathbf{A} = \{A_1, A_2, \ldots, A_m\}$ and $\mathbf{Q} = \{r_{t_a}(A_i, A_j), r_{t_a}(A_i, A_k), r_{t_b}(A_k, A_l) \ldots r_{t_z}(A_k, A_m)\}$. Here $t_a, t_b, t_z \in \tau$ are different relationship types between individuals (e.g. son, daughter, father, aunt). Our goal is to recover $\mathbf{F}$ from a set of k participant reports, $\mathcal{R}$.

We define these reports as $\mathcal{R} = \{\mathbf{R}^1, \mathbf{R}^2, \ldots, \mathbf{R}^k\}$, where superscripts will henceforth denote the participant associated with the reported data. Each report, $\mathbf{R}^i = \langle p^i, \mathbf{M}^i, \mathbf{Q}^i \rangle$ is defined by the reporting participant, p^i, the set of family members mentioned in the report, $\mathbf{M}^i$, and the participant's relationships to each mention, $\mathbf{Q}^i$. We denote the mentions, $\mathbf{M}^i = \{p^i, m_1^i, \ldots, m_{l_i}^i\}$, where each of the l_i mentions includes (possibly erroneous) personal attributes and corresponds to a distinct, unknown actor in the family tree (note that the participant is a mention as well). We denote the relationships $\mathbf{Q}^i = \{r_{t_a}(p^i, m_x^i), \ldots, r_{t_b}(p^i, m_y^i)\}$, where $t_a, t_b \in \tau$ denote the types of relation, and m_x^i and m_y^i denote the mentioned family members with whom the participant p^i shares the relation types t_a and t_b respectively. A participant p^i can have an arbitrary number of relations of the same type (e.g. two daughters, three brothers, zero sisters). Our goal is to examine all the mentions (participants and non-participants) and perform a matching across reports to create sets of mentions that correspond to the same actor. The ultimate task is to construct the unified family $\mathbf{F}$ from the collection of matches.

Entity Resolution Task. A prevalent approach to entity resolution is to cast the problem as a *binary, supervised*

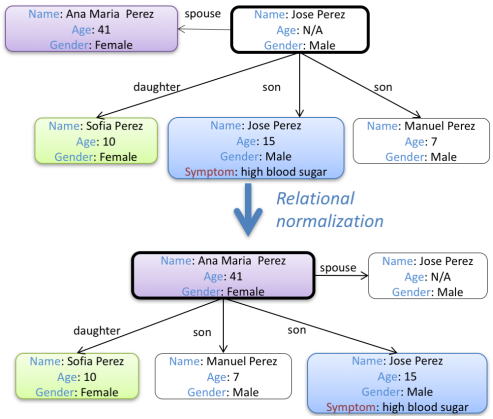

Fig. 2: Above: The tree corresponding to a participant report provided by "Jose Perez". Below: The derived normalized tree from the perspective of "Ana Maria Perez".

classification task and use machine learning to label each pair of entities as matching or non-matching. In our specific problem setting, this corresponds to introducing a variable $\textsc{Same}(x, y)$ for each pair of entities x, y occurring in distinct participant reports. Formally, we define $\forall_{i \neq j} \forall_{m_x^i \in \mathbf{M}^i} \forall_{m_y^j \in \mathbf{M}^j} \textsc{Same}(m_x^i, m_y^j)$. Our goal is to determine for each pair of mentions if they refer to the same actor.

In order to achieve this goal, we must learn a decision function that, given two mentions, determines if they are the same. Although the general problem of entity resolution is well-studied, we observe that a significant opportunity in this specific problem setting is the ability to leverage the familial relationships in each report to perform relational entity resolution. Unfortunately, the available reports, $\mathcal{R}$ are each provided from the perspective of a unique participant. This poses a problem since we require relational information for each *mention* in a report, not just for the reporting participant. We refer to the problem of recovering mention-specific relational features from participant reports as *relational normalization*, and present our algorithm in the next section.

III. Preprocessing via Relational Normalization

Since the relational information available in participant reports is unsuitable for entity resolution, we undertake the process of normalization to generate mention-specific relational information. To do so, we translate the relational information in a report $\mathbf{R}^i$ into an *ego-centric tree*, $\mathbf{T}_j^i$, for each mention m_j^i. Here the notation $\mathbf{T}_j^i$ indicates that the tree is constructed from the perspective of the j^{th} mention of the i^{th} report. We define $\mathbf{T}_j^i = \langle m_j^i, \mathbf{Q}_j^i \rangle$, where $\mathbf{Q}_j^i$ is a set of relationships. Constructing these trees consists of two steps: relationship inversion and relationship imputation.

a) Relationship Inversion: The first step in populating the ego-centric tree for m_j^i is to invert the relationships in $\mathbf{R}^i$ so that the first argument (subject) is m_j^i. More formally, for each relation type $t_j \in \tau$ such that $r_{t_j}(p^i, m_j^i)$, we introduce an inverse relationship $r_{t_i'}(m_j^i, p^i)$. In order to do so, we introduce a function $inverse(\tau, m_j^i, p^i) \rightarrow \tau$ which returns the appropriate inverse relationship for each relation type. Note that the inverse of a relation depends both on the mention and the participant, since in some cases mention attributes (e.g. father to daughter) or participant attributes (e.g. daughter to father) are used to determine the inverse.

b) Relationship Imputation: The next step in populating $\mathbf{T}_j^i$ is to impute relationships for m_j^i mediated through p^i. We define a function $impute(r_x(p^i, m_j^i), r_y(p^i, m_k^i)) \rightarrow r_k(m_j^i, m_k^i)$. For example, given the relations $\{r_{father}(p^i, m_j^i), r_{mother}(p^i, m_k^i)\}$ in $\mathbf{T}^i(p^i)$, then we impute the relations $r_{spouse}(m_j^i, m_k^i)$ in $\mathbf{T}_j^i$ as well as $r_{spouse}(m_k^i, m_j^i)$ in $\mathbf{T}_k^i$.

Figure 2 shows an example of the normalization process. We begin with the top tree centered on "Jose Perez" and after applying inversion and imputation we produce the bottom tree centered on "Ana Maria Perez". Finally, we note that since initially we have relational information for just one person in each tree, then it will be impossible to use any relational information if we do not perform the normalization step.

IV. Entity Resolution Model for Familial Networks

After recovering the mention-specific relational features from participant reports, our next step is to develop a model that is capable of collectively inferring mention equivalence using the attributes, diverse relational evidence, and logical constraints. We cast this entity resolution task as inference in a graphical model, and use the probabilistic soft logic (PSL) framework to define a probability distribution over co-referent mentions. Several features of this problem setting necessitate the choice of PSL: (1) entity resolution in familial networks is inherently collective, requiring constraints such as transitivity and bijection; (2) the multitude of relationship types require an expressive modeling language; (3) similarities between mention attributes take continuous values; (4) potential matches scale polynomially with mentions, requiring a scalable solution. PSL provides collective inference, expressive relational models defined over continuously-valued evidence, and formulates inference as a scalable convex optimization. In this section we provide a brief primer on PSL and then introduce our PSL model for entity resolution in familial networks.

A. Probabilistic Soft Logic (PSL)

Probabilistic soft logic is a probabilistic programming language that uses a first-order logical syntax to define a graphical model [2]. In contrast to other approaches, PSL uses continuous random variables in the $[0, 1]$ unit interval and specifies factors using convex functions, allowing tractable and efficient inference. PSL defines a Markov random field associated with a conditional probability density function over random variables $\mathbf{Y}$ conditioned on evidence $\mathbf{X}$,

$$P(\mathbf{Y}|\mathbf{X}) \propto \exp\Big(-\sum_{j=1}^{m} w_j \phi_j(\mathbf{Y}, \mathbf{X}) \Big) , \qquad (1)$$

where ϕ_j is a convex potential function and w_j is an associated weight which determines the importance of ϕ_j in the model. The potential ϕ_j takes the form of a *hinge-loss*:

$$\phi_j(\mathbf{Y}, \mathbf{X}) = (max\{0, \ell_j(\mathbf{X}, \mathbf{Y})\})^{p_j} . \qquad (2)$$

Here, ℓ_j is a linear function of $\mathbf{X}$ and $\mathbf{Y}$, and $p_j \in \{1, 2\}$ optionally squares the potential, resulting in a *squared-loss*. The resulting probability distribution is log-concave in $\mathbf{Y}$, so we can solve maximum a posteriori (MAP) inference exactly via convex optimization to find the optimal $\mathbf{Y}$. The convex formulation of PSL is the key to efficient, scalable inference in models with many complex interdependencies.

PSL derives the objective function by translating logical rules specifying dependencies between variables and evidence into hinge-loss functions. PSL achieves this translation by using the *Lukasiewicz* norm and co-norm to provide a relaxation of Boolean logical connectives [17]:

$$\begin{aligned} p \wedge q &= \max(0, p + q - 1) \\ p \vee q &= \min(1, p + q) \\ \neg p &= 1 - p \ . \end{aligned}$$

To illustrate PSL in an entity resolution context, the following rule encodes that mentions with similar names and the same gender might be the same person:

$$\textsc{SimName}(\mathtt{m}_1, \mathtt{m}_2) \wedge \textsc{eqGender}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ , \tag{3}$$

where $\textsc{SimName}(\mathtt{m}_1, \mathtt{m}_2)$ is a continuous observed atom taken from the string similarity between the names of $\mathtt{m}_1$ and $\mathtt{m}_2$, $\textsc{eqGender}(\mathtt{m}_1, \mathtt{m}_2)$ is a binary observed atom that takes its value from the logical comparison $\mathtt{m}_1.\text{gender} = \mathtt{m}_2.\text{gender}$ and $\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2)$ is a continuous value to be inferred, which encodes the probability that the mentions $\mathtt{m}_1$ and $\mathtt{m}_2$ are the same person. If this rule was instantiated with the assignments $\mathtt{m}_1$=`John Smith`, $\mathtt{m}_2$=`J Smith` the resulting hinge-loss potential function would have the form:

$$\begin{aligned} &\max(0, \textsc{SimName}(\texttt{John Smith}, \texttt{J Smith}) \\ &+ \textsc{eqGender}(\texttt{John Smith}, \texttt{J Smith}) \\ &- \textsc{Same}(\texttt{John Smith}, \texttt{J Smith}) - 1) \ . \end{aligned}$$

B. PSL Model

We define our model using rules similar to those in (3), allowing us to infer the SAME relation between mentions. Each rule encodes graph-structured dependency relationships drawn from the familial network (e.g., if two mentions are co-referent then their mothers should also be co-referent) or conventional attribute-based similarities (e.g., if two mentions have similar first and last name then they are possibly co-referent). We present a set of representative rules for our model, but note that additional features (e.g., locational similarity, conditions from a medical history, or new relationships) can easily be incorporated into our model with additional rules.

1) Name Similarity Rules: One of the most important mention attributes are mention names, and historically entity resolution research has focused on engineering similarity functions that accurately capture patterns in name similarity. In our model, we use two popular similarity functions, the Levenshtein [21] and Jaro-Winkler [27]. The first is known to work well for common typographical errors, while the second is specifically designed to work well with names. We introduce a rule that captures the intuition that when two mentions have similar names (according to the Jaro-Winkler similarity function) they are more likely to represent the same person:

$$\textsc{SimName}_{JW}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ .$$

This rule reinforces an important aspect of PSL: atoms take truth values in the $[0, 1]$ interval, capturing the degree of certainty of the inference. In the above rule, high name similarity results in greater confidence that two mentions are the same. However, we also wish to penalize pairs of mentions with dissimilar names from matching, for which we introduce the rule using the logical not ($\neg$):

$$\neg\textsc{SimName}_{JW}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ .$$

While we present these rules for a generic SIMNAME similarity function, our model introduces several name similarities for first, last, and middle names (we do not assume last names are the same across mentions). Similarly, our model can easily support alternative similarity metrics, such as Monge Elkan or Soundex [27], or similarities on combinations of names (e.g., first, middle, last).

2) Personal Information Similarity Rules: Beyond the name attributes of a mention, there are often additional attributes provided in reports that are useful for matching. For example, age is an important feature for entity resolution in family trees, since individuals may share a name across different generations. We introduce the following rule for age:

$$\textsc{SimAge}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \quad \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ .$$

The predicate $\textsc{SimAge}(\mathtt{m}_1, \mathtt{m}_2)$ takes values in the interval $[0, 1]$ and is computed as the ratio of the smallest over the largest value. While attributes like age have influence in matching, other attributes cannot be considered as evidence to matching but they are far more important in disallowing matches between the mentions. For example, same gender cannot be an indicator that two mentions are co-referent, however, different gender is a strong evidence that two mentions are not co-referent. To this end, we introduce rules that prevent mentions from matching when attributes differ:

$$\begin{aligned} \neg\textsc{SimAge}(\mathtt{m}_1, \mathtt{m}_2) &\Rightarrow \quad \neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \\ \neg\textsc{eqGender}(\mathtt{m}_1, \mathtt{m}_2) &\Rightarrow \quad \neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \\ \neg\textsc{eqLiving}(\mathtt{m}_1, \mathtt{m}_2) &\Rightarrow \quad \neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ . \end{aligned}$$

We note that the predicates $\textsc{eqGender}(\mathtt{m}_1, \mathtt{m}_2)$ and $\textsc{eqLiving}(\mathtt{m}_1, \mathtt{m}_2)$ have binary-valued atoms.

3) Relational Similarity Rules: Although attribute similarities provide useful features for entity resolution, in problem settings such as familial networks, relational features are necessary for matching. Relational features can be introduced in a multitude of ways. One possibility is to incorporate purely structural features, such as the number and types of relationships for each mention. For example, given a mention with two sisters and three sons and a mention with three sisters and three sons, we could design a similarity function for these relations. However, practically this approach lacks discriminative power because there are often mentions that have similar relational structures (e.g., having a mother) that refer to different entities. To overcome the lack of discriminative power, we augment

structural similarity with a matching process. For relationship types that are surjective, such as mother or father, the matching process is straightforward. We introduce a rule:

$$\textsc{SimMother}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ .$$

SIMMOTHER may have many possible definitions. In this work, SIMMOTHER is equal to the maximum of the Levenshtein and Jaro-Winkler similarities of the first names. However, when a relationship type is multi-valued, such as sister or son, a more sophisticated matching of the target individuals is required. Given a relation type t and possibly co-referent mentions m_1^i, m_2^j, we find all entities $M_x = \{m_x^i : r_t(m_1^i, m_x^i) \in \mathbf{Q}_1^i\}$ and $M_y = \{m_y^j : r_t(m_2^j, m_y^j) \in \mathbf{Q}_2^j\}$. Now we must define a similarity for the sets M_x and M_y, which in turn will provide a similarity for m_1^i and m_2^j. The similarity function we use is:

$$\textsc{Sim}_t(\mathtt{m}_1, \mathtt{m}_2) = \frac{1}{|M_x|} \sum_{\mathtt{m}_x \in M_x} \max_{\mathtt{m}_y \in M_y} \textsc{SimName}(\mathtt{m}_x, \mathtt{m}_y) \ .$$

For each $\mathtt{m}_x$ (an individual with relation t to $\mathtt{m}_1$), this computation greedily chooses the best $\mathtt{m}_y$ (an individual with relation t to $\mathtt{m}_2$). In our computation, we assume (without loss of generality, assuming symmetry of the similarity function) that $|M_x| < |M_y|$. While many possible similarity functions can be used for SIMNAME, we take the maximum of the Levenshtein and Jaro-Winkler similarities of the first names in our model.

Our main goal in introducing these relational similarities is to incorporate relational evidence that is compatible with simpler, baseline models. While more sophisticated than simple structural matches, these relational similarities are much less powerful than the transitive relational similarities supported by PSL, which we introduce in the next section.

4) Transitive Relational (Similarity) Rules: The rules that we have investigated so far can capture personal and relational similarities but they cannot identify similar persons in a collective way. To make this point more clear, consider the following observation: when we have high confidence that two persons are the same, we also have a stronger evidence that their associated relatives, e.g., father, are also the same. We encode this intuition with rules of the following type:

$$\textsc{Rel}(\mathtt{Father}, \mathtt{m}_1, \mathtt{m}_a) \wedge \textsc{Rel}(\mathtt{Father}, \mathtt{m}_2, \mathtt{m}_b) \\ \wedge \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \textsc{Same}(\mathtt{m}_a, \mathtt{m}_b) \ .$$

The rule above works well with surjective relationships, since each person can have only one (biological) father. When the cardinality is larger, e.g., sister, our model must avoid inferring that all sisters of two respective mentions are the same. In these cases we use additional evidence, i.e., name similarity, to select the appropriate sisters to match, as follows:

$$\textsc{Rel}(\mathtt{Sister}, \mathtt{m}_1, \mathtt{m}_a) \wedge \ \textsc{Rel}(\mathtt{Sister}, \mathtt{m}_2, \mathtt{m}_b) \\ \wedge \textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \wedge \ \textsc{SimName}(\mathtt{m}_a, \mathtt{m}_b) \Rightarrow \textsc{Same}(\mathtt{m}_a, \mathtt{m}_b) \ .$$

Just as in the previous section, we compute SIMNAME by using the maximum of the Jaro-Winkler and Levenshtein similarities for first names. For relationships that are one-to-one we can also introduce negative rules which express the intuition that two different persons should be connected to different persons given a specific relationship. For example, for a relationship such as spouse, we can use a rule such as:

$$\textsc{Rel}(\mathtt{Spouse}, \mathtt{m}_1, \mathtt{m}_a) \wedge \textsc{Rel}(\mathtt{Spouse}, \mathtt{m}_2, \mathtt{m}_b) \\ \wedge \neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \Rightarrow \neg\textsc{Same}(\mathtt{m}_a, \mathtt{m}_b) \ .$$

However, introducing similar rules for one-to-many relationships is inadvisable. To understand why, consider the case where two siblings do not match, yet they have the same mother, whose match confidence should remain unaffected.

5) Bijection and Transitivity Rules: Our entity resolution task has several natural constraints across reports. The first is bijection, namely that a mention m_x^i can match at most one mention, m_y^j from another report. Before introducing the rule, we define the predicate FROMREPORT (abbreviated $\textsc{FR}(\mathtt{m}_i, \mathtt{R}_i)$) which filters individuals from a particular participant report (e.g., $m_x^i \in \mathbf{M}^i$). According to the bijection rule, if mention m_a from report R_1 is matched to mention m_b from report R_2 then m_1 cannot be matched to any other mention from report R_2:

$$\textsc{FR}(\mathtt{m}_a, \mathtt{R}_1) \wedge \textsc{FR}(\mathtt{m}_b, \mathtt{R}_2) \wedge \textsc{FR}(\mathtt{m}_c, \mathtt{R}_2) \\ \wedge \textsc{Same}(\mathtt{m}_a, \mathtt{m}_b) \Rightarrow \neg\textsc{Same}(\mathtt{m}_a, \mathtt{m}_c) \ .$$

Note that this bijection is *soft*, and does not guarantee a single, exclusive match for $\mathtt{m}_a$, but rather attenuates the confidence in each possible match modulated by the evidence for the respective matches. A second natural constraint is transitivity, which requires that if m_a^i and m_y^j are the same, and mentions m_y^j and m_c^k are the same, then mentions m_a^i and m_c^k should also be the same. We capture this constraint as follows:

$$\textsc{FR}(\mathtt{m}_a, \mathtt{R}_1) \wedge \textsc{FR}(\mathtt{m}_b, \mathtt{R}_2) \wedge \textsc{FR}(\mathtt{m}_c, \mathtt{R}_3) \\ \wedge \textsc{Same}(\mathtt{m}_a, \mathtt{m}_b) \wedge \textsc{Same}(\mathtt{m}_b, \mathtt{m}_c) \Rightarrow \textsc{Same}(\mathtt{m}_a, \mathtt{m}_c) \ .$$

6) Prior Rule: Entity resolution is typically an imbalanced classification problem, meaning that most of the mention pairs are not co-referent. We can model our general belief that two mentions are likely not co-referent, using the prior rule:

$$\neg\textsc{Same}(\mathtt{m}_1, \mathtt{m}_2) \ .$$

7) Flexible Modeling: We reiterate that in this section we have only provided *representative* rules used in our PSL model for entity resolution. Moreover, a key feature of our model is the flexibility and the ease with which it can be extended to incorporate new features. For example, adding additional attributes, such as profession or location, is easy to accomplish following the patterns of Subsection IV-B2. Incorporating additional relationships, such as cousins or friends is simply accomplished using the patterns in Subsections IV-B3 and IV-B4. Our goal has been to present a variety of patterns that are adaptable across different datasets and use cases.

C. Learning the PSL Model

Given the above model, we use observational evidence (similarity functions and relationships) and variables (potential matches) to define a set of ground rules. Each ground rule is translated into a hinge-loss potential function of the form (2) defining a Markov random field, as in (1) (Section IV-A). Then, given the observed values $\mathbf{X}$ our goal is to find the

most probable assignment to the unobserved variables $\mathbf{Y}$ by performing joint inference over interdependent variables.

As we discussed in IV-A, each of the first-order rules introduced in the previous section is associated with a non-negative weight w_j in Equation 1. These weights determine the relative importance of each rule, corresponding to the extent to which the corresponding hinge function ϕ_j alters the probability of the data under Equation 1. A higher weight w_j corresponds to a greater importance of information source j in the entity resolution task. We learn rule weights using Bach et al.'s [3] approximate maximum likelihood weight learning algorithm, using a held-out training set. Finally, since the output of the PSL model is a soft-truth value for each pair of mentions, to evaluate our matching we choose a threshold to make a binary match decision. We choose the optimal threshold on a held-out development set to maximize the F-measure score, and use this threshold when classifying data in the test set.

D. Satisfying Matching Restrictions

One of the key constraints in our model is a bijection constraint that requires that each mention can match at most one mention in another report. Since the bijection rule in PSL is *soft*, in some cases, we may get multiple matching mentions for a report. To enforce this restriction, we introduce a greedy 1:1 matching step. We use a simple algorithm that first sorts output matchings by the truth value of the $\textsc{Same}(m_x^i, m_y^j)$ predicate. Next, we iterate over this sorted list of mention pairs, choosing the highest ranked pair for an entity, (m_x^i, m_y^j). We then remove all other potential pairs, $\forall_{m_a^i, a \neq x}(m_a^i, m_y^j)$ and $\forall_{m_b^j, b \neq y}(m_x^i, m_b^j)$, from the matching. This approach is simple to implement, efficient, and can potentially improve model performance, as we will discuss in our experiments.

V. Experimental Validation

A. Datasets and Baseline

For our experimental evaluation we use two datasets, a clinical dataset provided by the National Institutes of Health (NIH) [12] and a public dataset crawled from the structured knowledge repository, Wikidata.[1] We provide summary statistics for both datasets in Table I.

The NIH dataset was collected by interviewing 497 patients from 162 families and recording family medical histories. For each family, 3 or 4 patients were interviewed, and each interview yielded a corresponding ego-centric view of the family tree. Patients provided first and second degree relations, such as parents and grandparents. In total, the classification task requires determining co-reference for about $300,000$ pairs of mentions. The provided dataset was manually annotated by at least two coders, with reconciliation of differences. Only 1.6% of the potential pairs are co-referent, resulting in a severely imbalanced classification, which is common in entity resolution scenarios.

The Wikidata dataset was generated by crawling part of the Wikidata[2] knowledge base. More specifically, we generated a seed set of 419 well-known politicians or celebrities, e.g., "Barack Obama".[3] For each person in the seed set, we retrieved attributes from Wikidata including their full name (and common variants), age, gender, and living status. Wikidata provides familial data only for first-degree relationships, i.e., siblings, parents, children, and spouses. Using the available relationships, we also crawled Wikidata to acquire attributes and relationships for each listed relative. This process resulted in 419 families. For each family, we have a different number of family trees (ranging from 2 to 18) with $1,844$ family trees in total, and $175,000$ pairs of potentially co-referent mentions (8.7% of which are co-referent). Mentions in Wikidata are associated with unique identifiers, which we use as ground truth. In the next section, we describe how we add noise to this dataset to evaluate our method.

Dataset	NIH	Wikidata
No. of families	162	419
No. of family trees	497	1,844
No. of mentions	12,111	8,553
No. of 1^{st} degree relationships	46,983	49,620
No. of 2^{nd} degree relationships	67,540	0
No. of pairs for comparison	300,547	174,601
% of co-referent pairs	1.6%	8.69%

TABLE I: Datasets description

We compare our approach to state-of-the-art classifiers that are capable of providing the probability that a given pair of mentions is co-referent. Probability values are essential since they are the input to the greedy 1-1 matching restrictions algorithm. We compare our approach to the following classifiers: logistic regression (LR), logistic model trees (LMTs), and support vector machines (SVMs). For LR we use a multinomial logistic regression model with a ridge estimator [5] using the implementation and improvements of WEKA [11] with the default settings. For LMTs we use Weka's implementation [19] with the default settings. For SVMs we use Weka's LibSVM library [6], along with the functionality to estimate probabilities. To select the best SVM model we follow the process described by Hsu et al. [15]: we first find the kernel that performs best, which in our case was the radial basis function (RBF). We then perform a grid search to find the best values for C and γ parameters. The starting point for the grid search was the default values given by Weka, i.e., C=1 and γ=1/(number of attributes), and we continue the search with exponentially increasing/decreasing sequences of C and γ. We note however that, unlike our model, none of these off-the-shelf classifiers can incorporate transitivity or bijection.

B. Experimental Setup

We evaluate our entity resolution approach using the metrics of *precision*, *recall*, and *F-measure* for the positive (co-referent) class which are typical for entity resolution problems [7]. For all reported results we use 5-fold cross-validation, with distinct training, development, and test sets. Folds are generated by randomly assigning each of the 162 (NIH) and 419 (Wikidata) families to one of five par-

[1]Code and data available at: https://github.com/pkouki/icdm2017.

[2]https://www.wikidata.org/

[3]https://www.wikidata.org/wiki/Q76

titions, yielding folds that contain the participant reports for approximately 32 (NIH) and 83 (Wikidata) familial networks.

The NIH dataset is collected in a real-world setting where information is naturally incomplete and erroneous, and attributes alone are insufficient to resolve the entities. However, the Wikidata resource is heavily curated and assumed to contain no noise. To simulate the noisy conditions of real-world datasets, we introduced additive Gaussian noise to the similarity scores. Noise was added to each similarity metric described in the previous section (e.g., first name Jaro-Winkler, age ratio). In our full experiments we considered varying levels of noise, finding higher noise correlated with lower performance. Due to space limitations, results are presented only for noise terms drawn from a $N(0, 0.16)$ distribution.

In each experiment, for PSL, we use three folds for training the model weights, one fold for choosing a binary classification threshold, and one fold for evaluating model performance. To train the weights, we use PSL's default values for the two parameters: number of iterations (equal to 25) and step size (equal to 1). For SVMs, we use three folds for training the SVMs with the different values of C and γ, one fold for choosing the best C and γ combination, and one fold for evaluating model performance. For LR and LMTs we use three folds for training the models with the default parameter settings and one fold for evaluating the models. We train, validate, and evaluate using the same splits for all models. We report the average precision, recall, and F-measure together with the standard deviation across folds.

C. Experiments

For our PSL model, we start with a simple feature set using only name similarities (see Subsection IV-B1), transitivity and bijection soft constraints (see Subsection IV-B5), and a prior (see Subsection IV-B6). We progressively enhance the model by adding attribute similarities computed based on personal information, relational similarities, and transitive relationships. Finally, since our dataset poses the constraint that each person from one report can be matched with at most one person from another report, we consider only solutions that satisfy this constraint. To ensure that the output is a valid solution, we apply the greedy 1:1 matching restriction algorithm (see Subsection IV-D) on the output of the each model. For each of the experiments we also ran baseline models that use the same information as the PSL models in the form of features. Unlike our models implemented within PSL, the models from the baseline classifiers do not support collective reasoning, i.e., applying transitivity and bijection is not possible in the baseline models. However, we are able to apply the greedy 1:1 matching restriction algorithm on the output of each of the classifiers for each of the experiments to ensure that we provide a valid solution. We ran the following experiments:

Names: A PSL model with rules only on name similarities, as discussed in Section IV-B1. We also ran LR, LMTs, and SVMs models that use as features the first, middle, and last name similarities based on Levenshtein and Jaro-Winkler measures.

Names + Personal Info: We enhance **Names** by adding rules about personal information similarities, as discussed in Section IV-B2. For the baselines, we add corresponding features for age similarity, gender, and living status. This is the most complex feature set that can be supported without using the normalization procedure we introduce in Section III.

Names + Personal + Relational Info (1^{st} degree): For this model and all subsequent models we perform normalization to enable the use of relational evidence for entity resolution. We present the performance of two PSL models. In the first model, PSL(R_1), we add first degree relational similarity rules, as discussed in Section IV-B3. First degree relationships are: mother, father, daughter, son, brother, sister, spouse. In the second model, PSL(R_1TR_1), we extend the PSL(R_1) by adding first-degree transitive relational rules, as discussed in Section IV-B4. For the baselines, we extend the previous models by adding first-degree relational similarities as features. However, it is not possible to include features similar to the transitive relational rules in PSL, since these models do not support collective reasoning.

Names + Personal + Relational Info (1^{st} + 2^{nd} degree): As above, we evaluate the performance of two PSL models. In the first experiment, PSL($R_{12}TR_1$), we enhance the model PSL(R_1TR_1) by adding second-degree relational similarity rules, as discussed in Section IV-B3. Second degree relationships are: grandmother, grandfather, granddaughter, grandson, aunt, uncle, niece, nephew. In the second experiment, PSL($R_{12}TR_{12}$), we enhance PSL($R_{12}TR_1$) by adding second-degree transitive relational similarity rules, as discussed in Section IV-B4. For the baselines, we add the second-degree relational similarities as features. Again, it is not possible to add features that capture the transitive relational similarity rules. Since Wikidata dataset does not provide second degree relations, we do not report experimental results for this case.

D. Discussion

We present our results in Table II. For each experiment, we denote with bold the best performance in terms of the F-measure. We present the results for both our method and the baselines and only for the positive class (co-referent entities). Due to the imbalanced nature of the task, performance on non-matching entities is similar across all approaches, with precision varying from 99.6% to 99.9%, recall varying from 99.4% to 99.9%, and F-measure varying from 99.5% to 99.7% for the NIH dataset. For the Wikidata, precision varies from 98.7% to 99.8%, recall varies from 98.9% to 99.9%, and F-measure varies from 99.5% to 99.7%. Next, we summarize some of our insights from the results of Table II.

PSL models universally outperform baselines: In each experiment PSL outperforms all the baselines using the same feature set. With one exception (for NIH, **Names + Personal Info**), PSL produces a statistically significant improvement in F-measure as measured by a paired t-test with $\alpha = 0.05$. Of the baselines, LMTs perform best in all experiments and will be used for illustrative comparison. When using name similarities only (**Names** models in Table II) PSL outperforms LMTs by 2.3% and 3.5% (absolute value) for the NIH and the Wikidata dataset accordingly. When adding personal information similarities (**Names + Personal Info**), PSL outperforms LMTs by

Method		NIH			Wikidata		
		Precision(SD)	**Recall(SD)**	**F-measure(SD)**	**Precision(SD)**	**Recall(SD)**	**F-measure(SD)**
Names	LR	0.871 (0.025)	0.686 (0.028)	0.767 (0.022)	0.905 (0.015)	0.598 (0.022)	0.720 (0.018)
	SVMs	0.870 (0.022)	0.683 (0.027)	0.765 (0.020)	0.941 (0.017)	0.607 (0.034)	0.738 (0.026)
	LMTs	0.874 (0.020)	0.717 (0.027)	0.787 (0.022)	0.926 (0.011)	0.660 (0.034)	0.770 (0.023)
	PSL	0.866 (0.021)	0.761 (0.028)	**0.810** (0.023)*	0.870 (0.019)	0.751 (0.038)	**0.805** (0.020)*
Names + Personal Info	LR	0.968 (0.010)	0.802 (0.035)	0.877 (0.024)	0.953 (0.015)	0.713 (0.032)	0.815 (0.022)
	SVMs	0.973 (0.008)	0.832 (0.025)	0.896 (0.016)	0.970 (0.011)	0.723 (0.034)	0.828 (0.023)
	LMTs	0.966 (0.011)	0.859 (0.018)	0.909 (0.015)	0.960 (0.014)	0.745 (0.037)	0.838 (0.022)
	PSL	0.937 (0.016)	0.893 (0.019)	**0.915** (0.015)	0.909 (0.025)	0.813 (0.040)	**0.857** (0.017)*
Names + Personal + Relational Info (1^{st} degree)	LR	0.975 (0.011)	0.804 (0.035)	0.881 (0.025)	0.962 (0.013)	0.756 (0.028)	0.846 (0.015)
	SVMs	0.983 (0.008)	0.835 (0.026)	0.903 (0.018)	0.975 (0.012)	0.776 (0.035)	0.864 (0.019)
	LMTs	0.961 (0.013)	0.856 (0.028)	0.905 (0.020)	0.967 (0.015)	0.785 (0.037)	0.866 (0.019)
	PSL(R_1)	0.932 (0.013)	0.888 (0.030)	0.909 (0.018)	0.915 (0.017)	0.867 (0.029)	0.890 (0.010)
	PSL(R_1TR_1)	0.956 (0.006)	0.924 (0.024)	**0.940** (0.014)*	0.914 (0.016)	0.880 (0.018)	**0.896** (0.006)*
Names + Personal + Relational Info (1^{st} + 2^{nd} degree)	LR	0.970 (0.012)	0.807 (0.051)	0.880 (0.032)	-	-	-
	SVMs	0.985 (0.006)	0.856 (0.029)	0.916 (0.019)	-	-	-
	LMTs	0.975 (0.008)	0.872 (0.016)	0.921 (0.011)	-	-	-
	PSL($R_{12}TR_1$)	0.958 (0.005)	0.926 (0.021)	0.942 (0.014)	-	-	-
	PSL($R_{12}TR12$)	0.961 (0.011)	0.931 (0.020)	**0.946** (0.010)*	-	-	-

TABLE II: Performance of PSL and baseline classifiers with varying types of rules/features. Numbers in parenthesis indicate standard deviations. Bold shows the best performance in terms of F-measure for each feature set. Statistical significance at $\alpha = 0.05$ when using paired t-test is denoted by *.

0.6% and 1.9% for the NIH and the Wikidata accordingly, although the improvement for NIH is not statistically significant. For the experiment **Names + Personal + Relational Info** 1^{st} **degree**, the PSL model that uses both relational and transitive relational similarity rules, PSL(R_1TR_1), outperforms LMTs by 3.5% for the NIH and 3.0% for the Wikidata. Finally, for the NIH dataset, for the experiment that additionally uses relational similarities of second degree, the best PSL model, PSL($R_{12}TR_{12}$), outperforms LMTs by 2.5%.

Name similarities are not enough: When we incorporate personal information similarities (**Names + Personal Info**) on top of the simple **Names** model that uses name similarities only, we get substantial improvements for the PSL model, i.e., 10.5% in F-measure. The same observation is also true for all baseline models, with SVMs getting the most benefit out of the addition of personal information with an increase of 13.1%.

First-degree relationships help most in low noise scenarios: We found that reliable relational evidence improves performance, but noisy relationships can be detrimental. In the NIH dataset, incorporating first-degree relationships using the simple relational similarity function defined in Subsection IV-B3 decreases performance slightly for the PSL and the LMTs models (0.6% and 0.4% respectively). For LR and SVMs, F-measure increases slightly (0.4% and 0.7% respectively). However, for the Wikidata, the addition of simple relational similarities increased F-measure by 3.3% for PSL(R_1), 2.8% for LMTs, 3.6% for SVMs, and 3.1% for LR. We believe that the difference in the effect of the simple relational features is due to the different noise in the two datasets. NIH is a real-world dataset with incomplete and unreliable information, while Wikidata is considered to contain no noise. As a result, we believe that both the baseline and PSL models are able to cope with the artificially introduced noise, while it is much more difficult to deal with real-world noisy data.

Collective relations yield substantial improvements: When we incorporate collective, transitive relational rules to the PSL(R_1) model resulting to the PSL(R_1TR_1) model – a key differentiator of our approach – we observe a 3.1% improvement in F-measure for the NIH dataset which is a result of an increase of 3.6% for the recall and 2.4% for the precision. Adding collective rules allows decisions to be propagated between related pairs of mentions, exploiting statistical signals across the familial network to improve recall. The Wikidata also benefits from collective relationships, but the 0.6% improvement in F-measure score is much smaller. For this cleaner dataset, we believe that simple relational similarity rules were informative enough to dampen the impact of transitive relational similarity rules. As a result, these rules are not as helpful as in the more noisy NIH dataset.

Second-degree similarities improves performance: The addition of simple relational similarities from second degree relationships, such as those available in the NIH dataset, yield improvements in all models except LR. For our approach, PSL($R_{12}TR_1$), slightly improves the PSL(R_1TR_1) model (0.2% for F-measure), while the addition of second-degree transitive relational features (model PSL($R_{12}TR_{12}$)) further improves slightly the performance by 0.4%. When adding second-degree relationships, we observe a pronounced increase in the F-measure for two baselines (1.6% for LMTs and 1.3% for SVMs), while LR has a small drop of 0.1%.

Precision-recall balance can change using a different criterion for the threshold value: As we discussed in Section IV-C for the PSL model we choose the optimal threshold to maximize the F-measure score. This learned threshold achieves a precision-recall balance that favors recall at the expense of precision. For both datasets, our model's recall is significantly higher than all the baselines in all the experiments. However, since PSL outputs soft truth values, changing the threshold selection criteria in response to the application domain (e.g., prioritizing cleaner matches over coverage) can allow the model to emphasize precision over recall.

Matching restrictions always improves F-measure: We note

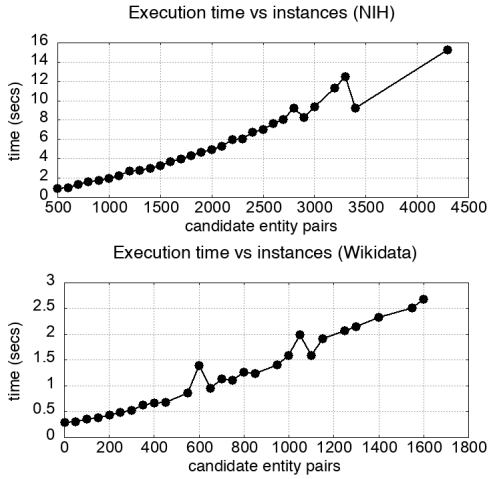

Fig. 3: An analysis of the scalability of our system. As the number of potentially co-referent entity pairs increases, the execution time of our model grows linearly for both datasets.

that valid solutions in our entity resolution setting require that an entity matches at most one entity in another ego-centric network. To enforce this restriction, we apply a 1-1 matching algorithm on the raw output of all models (Section IV-D). Applying matching restrictions adjusts the precision-recall balance of all models. For both PSL and the baselines across both datasets, when applying the 1-1 matching restriction algorithm, we observe a sizable increase in precision and a marginal drop in recall. This pattern matches our expectations, since the algorithm removes predicted co-references (harming recall) but is expected to primarily remove false-positive pairs (helping precision). Overall, the application of the 1-1 matching restrictions improves the F-measure for all algorithms and all datasets. Since the results before the 1-1 matching do not represent valid solutions and it is not straightforward to compare across algorithms we do not report them here.

PSL scales linearly with instances: One motivation for choosing PSL to implement our entity resolution model was the need to scale to large datasets. To validate the scalability of our approach, we vary the number of instances, consisting of pairs of candidate co-referent entities, and measure the execution time of inference. In Figure 3 we plot the average execution time relative to the number of candidate entity pairs. Our results indicate that our model scales almost linearly with respect to the number of comparisons. For the NIH dataset, we note one prominent outlier, for a family with limited relational evidence resulting in lower execution time. Conversely, for the Wikidata, we observe two spikes which are caused by families that contain relatively dense relational evidence compared to similar families. We finally note that we expect these scalability results to hold as the datasets get bigger since the execution time depends on the number of comparisons and the number of relations per family.

VI. Related Work

There is a large body of prior work in the general area of entity resolution [7]. In this work we propose a collective approach that makes extensive use of relational data. In the following we review collective relational entity resolution approches which according to Rastogi et al. [24] can be either iterative or purely-collective.

For the iterative collective classification case, Bhattacharya and Getoor [4] propose a method based on greedy clustering over the relationships. This work considers only one single relation type, while we consider several types. Dong et al. [9] propose another iterative approach which combines contextual information with similarity metrics across attributes. In our approach, we perform both reference and relation enrichment, by applying inversion and imputation. Finally, Kalashnikov and Mehrotra [16] propose an approach for the reference disambiguation problem where the entities are already known. In our case, we do not know the entities beforehand.

In the case of purely collective approaches, Arasu et al. [1] propose the Dedupalog framework for collective entity resolution with both soft and hard constraints. Dedupalog is well-suited for datasets having the need to satisfy several matching restrictions. In our case, we have several soft rules with a smaller number of constraints. In another approach, Culotta and McCallum [8] design a conditional random field model incorporating relationship dependencies. In this work too, the number of relationship types considered is small. Finally, Singla and Domingos [25] combine first-order logic and Markov random fields to perform collective classification. The proposed Markov Logic Networks (MLNs) operate on undirected graphical models using a first-order logic as their template language, like PSL. However, the predicates take only boolean values, while in PSL the predicates take soft truth values in the range $[0, 1]$ which is more appropriate for representing notions such as name similarities. Additionally, according to related work [3], HL-MRFs can achieve improved performance in much less time compared to MLNs.

Overall, the purely collective approaches come with a high computational cost for performing probabilistic inference. As a result, they cannot scale to large datasets unless we use techniques that make the EM algorithm scalable [24]. Our approach uses PSL which ensures scalable and exact inference by solving a convex optimization problem in parallel. Speed and scalability is of paramount importance in entity resolution and in particular when we run the prediction task collectively using transitivity and bijection rules.

Regarding the problem of entity resolution in familial networks, we recently proposed a first approach [18]. The problem setting is the same as in the current work, but the approach is non-collective using well-studied classifiers enhanced with features capturing relational similarity. In this work we propose a more sophisticated collective approach to the familial entity resolution problem.

Additionally, there are some works from the ontology alignment and knowledge graph identification domains that are close to our approach. Suchanek et al. [26] propose a probabilistic approach for ontology alignment. The tool accepts as input two ontologies and distinguishes the same relations, classes, and instances. As a result, the approach does not take into account transitivity and bijection constraints,

which are key features in the familial networks. Finally, Pujara and Getoor [23] use PSL to design a general mechanism for entity resolution in knowledge graphs, a setting with a similarly rich relational structure. Their work considers entity resolution within and between graphs. However, familial networks have unique characteristics and constraints that differ substantially from knowledge graphs, and in particular they do not explicitly consider the problem of entity resolution across several subgraphs.

VII. Conclusions and Future Work

Entity resolution in familial networks poses several challenges, including heterogeneous relationships that introduce collective dependencies between decisions and inaccurate attribute values that undermine classical approaches. In this work, we propose a scalable collective approach based on probabilistic soft logic that leverages attribute similarities, relational information, and logical constraints. A key differentiator of our approach is the ability to support bijection and different types of transitive relational rules that can model the complex familial relationships. Moreover, our method is capable of using training data to learn the weight of different similarity scores and relational features, an important ingredient of relational ER. In our experimental evaluation, we demonstrated that our framework can effectively combine different signals, resulting in improved performance over state-of-the-art approaches on two datasets.

In this paper, we motivate the importance of our approach with an application for resolving mentions in healthcare records. However, the problem of entity resolution in richly structured domains has many additional applications. For example, many companies[4] provide genealogical discovery services, which require a similar entity resolution process. We also foresee applications in social networks, where the problem of linking user accounts across several social platforms in the presence of a diverse set of relationships (e.g., friends, followers, followees, family cycles, shared groups), ambiguous names, and collective constraints such as bijection and transitivity, provide a similar set of opportunities and challenges.

In future work, we plan to apply our approach to a broader set of problems and discuss general strategies for multirelational entity resolution. Additionally, we plan to explore structured output learning techniques [22] inside PSL. Such techniques can directly consider the matching constraints during the learning phase instead of post processing the classification results. We also plan to explore temporal relations, e.g. ex-wife, and more complex relationships, e.g. adopted child. Finally, in certain cases, we might inadvertently introduce inaccurate relations when following the approach of Section III. To address this, we plan to expand our work to account for uncertainty in the relational normalization step by assuming a probability assigned to each populated relationship instead of the hard values that we currently assign.

[4]ancestry.com, genealogy.com, familysearch.org

Acknowledgements

This work was partially supported by the National Science Foundation grant IIS1218488 and by the National Human Genome Research Institute Division of Intramural Research at the National Institutes of Health (ZIA HG2000397, Koehly PI). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the National Institutes of Health.

References

[1] A. Arasu, C. Ré, and D. Suciu. Large-scale deduplication with constraints using dedupalog. In *ICDE*, 2009.

[2] S. Bach, M. Broecheler, B. Huang, and L. Getoor. Hinge-loss markov random fields and probabilistic soft logic. *JMLR*, 2017.

[3] S. Bach, B. Huang, B. London, and L. Getoor. Hinge-loss Markov random fields: Convex inference for structured prediction. In *UAI*, 2013.

[4] I. Bhattacharya and L. Getoor. Collective entity resolution in relational data. *TKDD*, 1(1), 2007.

[5] S. Cessie and J. Houwelingen. Ridge estimators in logistic regression. *Applied Statistics*, 41(1), 1992.

[6] C. Chang and C. Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3), 2011.

[7] P. Christen. *Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection*. Springer, 2012.

[8] A. Culotta and A. McCallum. Joint deduplication of multiple record types in relational data. In *CIKM*, 2005.

[9] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD*, 2005.

[10] J. Efremova, B. Ranjbar-Sahraei, H. Rahmani, F. Oliehoek, T. Calders, K. Tuyls, and G. Weiss. *Multi-Source Entity Resolution for Genealogical Data*, pages 129–154. Population Reconstruction, 2015.

[11] E. Frank, M. Hall, and I. Witten. *The WEKA Workbench. Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2016.

[12] A. Goergen, S. Ashida, K. Skapinsky, H. de Heer, A. Wilkinson, and L. Koehly. Knowledge is power: Improving family health history knowledge of diabetes and heart disease among multigenerational mexican origin families. *Public Health Genomics*, 19(2), 2016.

[13] R. Hanneman and F. Riddle. *Introduction to Social Network Methods*. University of California, Riverside, 2005.

[14] K. Harron, A. Wade, R. Gilbert, B. Muller-Pebody, and H. Goldstein. Evaluating bias due to data linkage error in electronic healthcare records. *BMC Medical Research Methodology*, 14(36), 2014.

[15] C. Hsu, C. Chang, and C. Lin. *A Practical Guide to Support Vector Classification*. Technical report, Department of Computer Science, National Taiwan University, 2003.

[16] D. Kalashnikov and S. Mehrotra. Domain-independent data cleaning via analysis of entity-relationship graph. *TODS*, 31(2), 2006.

[17] A. Kimmig, S. Bach, M. Broecheler, B. Huang, and L. Getoor. A short introduction to probabilistic soft logic. In *NIPS Workshop on Probabilistic Programming*, 2012.

[18] P. Kouki, C. Marcum, L. Koehly, and L. Getoor. Entity resolution in familial networks. In *KDD, Workshop on Mining and Learning with Graphs*, 2016.

[19] N. Landwehr, M. Hall, and E. Frank. Logistic model trees. *Machine Learning*, 95(1-2), 2005.

[20] X. Li and C. Shen. Linkage of patient records from disparate sources. *Statistical Methods in Medical Research*, 22(1), 2008.

[21] G. Navarro. A Guided Tour to Approximate String Matching. *ACM Computing Surveys*, 33(1), 2001.

[22] S. Nowozin, P. Gehler, J. Jancsary, and C. Lampert. *Advanced Structured Prediction*. The MIT Press, 2014.

[23] J. Pujara and L. Getoor. Generic statistical relational entity resolution in knowledge graphs. In *IJCAI, Workshop on Statistical Relational AI*, 2016.

[24] V. Rastogi, N. Dalvi, and M. Garofalakis. Large-scale collective entity matching. In *VLDB*, 2011.

[25] P. Singla and P. Domingos. Entity resolution with Markov logic. In *ICDM*, 2006.

[26] F. Suchanek, S. Abiteboul, and P. Senellart. Paris: Probabilistic alignment of relations, instances, and schema. *Proc. VLDB Endow.*, 5(3), 2011.

[27] W. Winkler. Overview of record linkage and current research directions. Technical report, US Census Bureau, 2006.