Entity Resolution Using Inferred Relationships and Behavior

Jonathan Mugan, Ranga Chari, Laura Hitt,
Eric McDermid, Marsha Sowell, Yuan Qu
21CT, Inc., Austin, Texas, USA
Email: jmugan@21ct.com, rangarajan.chari@gmail.com,
lhitt@21ct.com, em4617@gmail.com,
msowell@21ct.com, yqu@21ct.com

Thayne Coffman
RGM Advisors, LLC
Austin, Texas USA
Email: tcoffman@rgmadvisors.com

***Abstract*—We present a method for entity resolution that infers relationships between observed identities and uses those relationships to aid in mapping identities to underlying entities. We also introduce the idea of using graphlets for entity resolution. Graphlets are collections of small graphs that can be used to characterize the "role" of a node in a graph. The idea is that graphlets can provide a richer set of features to characterize identities. We validate our method on standard author datasets, and we further evaluate our method using data collected from Twitter. We find that inferred relationships and graphlets are useful for entity resolution.**

I. Introduction

As our data collection abilities expand, we increasingly need the ability to bring multiple sources of information together into a coherent whole. The focus of this work is on entity resolution. Entity resolution is about determining when multiple observable identities correspond to the same underlying entity. Imagine a fraud ring devoted to stealing government benefits that authorities break up and then re-establishes itself under a new set of identities. How could we find this same network of people again?

Entity resolution can also be useful when data comes from different sources. We may have social media data of a set of Twitter users and a set of Facebook users, and by observing their behavior, we may want to know which Twitter users correspond to which Facebook users. Or, we may have satellite data tracking ships as well as AIS data tracking ships, and we want to be able to link the same ships together.

Entity resolution has traditionally focused on attribute matching. Attribute matching is an approach whereby one examines the attributes (features) of two identities to see if they match up. However, the events of the world are now increasingly represented in digital form, allowing us to see not only the subjects in which we are interested, but also the objects with which those subjects interact. This means that in addition to having access to the attributes of subjects, we can capture their relationships to other objects, providing a richer view of the environment. Fig. 1 shows an example of representing an entity as a vector of attributes and relationships.

Graduating from an attribute-based representation to a relational-based representation has three consequences for understanding data.

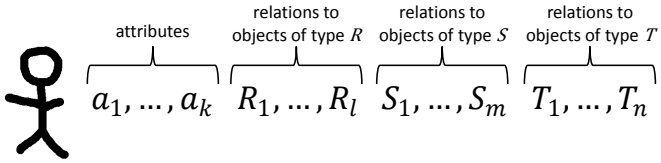


Fig. 1: Example relational representation. The subject (entity), shown as a stick figure, is represented as a set of attributes $a_1, \ldots, a_k$ and three different kinds of relations. The subject is related via relation R to l objects (other entities), via relation S to m objects, and via relation T to n objects.

1) One must consider both the subject and the objects to which it is related.
2) That object can also be the subject in multiple other relationships, so the initial subject is related to those objects via second degree.
3) The objects that a subject is related to also have attributes. For example, one could consider if the majority of the Twitter followers of an individual are interested in Chess.

Relational entity resolution has recently gained more attention, however the majority of this work on relational entity resolution assumes that one knows the relationships between the entities. There have been multiple efforts [1], [2], [3] in the social media domain that show how entities can be resolved using lists of followers or friends. However, for multiple applications in the real world, the picture is not so neat. Fraudsters dont publish lists of their co-conspirators—their conspirators must be inferred by observing their behavior. Inferring these conspirators requires more than looking at whom they communicate with, because that leads to data with too much noise.

In this work, we present a method that resolves entities where a list of conspirators is not given but must instead be inferred from behavior. We build on previous work of temporal group detection [4]. Temporal group detection analyzes communications over time and finds groups of individuals who consistently communicate. The first contribution of this paper is to use the relationships found through temporal group detection to help in performing entity resolution on relational data where the relations must be inferred.

In addition to defining relationships by observing behavior, we also want to use the behavior itself to help with entity resolution. The second contribution of this paper is incorporating graphlets into entity resolution. Graphlets are small graph fragments, and we use them to characterize the role that the subject plays in its environment.

The structure of this paper is as follows. Our work infers relationships from observed behavior, but we discuss related work in entity resolution when the relationships are given. We then provide the details of our method. Following that, we provide an evaluation of our method and we conclude.

II. Related Work

One of earliest efforts in relational entity resolution was Markov Logic Networks (MLNs) [5]. An MLN is a graph-based representation of a set of possible worlds. Nodes in MLNs represent ground atoms. Edges between nodes are present if they are related by a formula. Bhattacharya and Getoor [1] use an agglomerative clustering algorithm that considers attributes and relations among identities. A cluster is a set of identities that correspond to the same entity. Narayanan and Shmatikov [2] is very similar to Bhattacharya and Getoor [1]. They begin with a set of "seed matches similar to the result of the bootstrapping method of Bhattacharya and Getoor, and they then iteratively add resolutions based on the number of identically resolved neighbors between two nodes. Bartunov et al. [6] model the problem using a conditional random field and they solve it using quadratic optimization. Peled et al. [3] uses a supervised machine learning approach. With some entities labeled, they learn a model to determine if two entities in two social networks are the same. They collected data from Facebook and Xing (a network of business professionals centered in Europe).

In addition to the introduction of graphlets for entity resolution, our work differs from the work discussed here because instead of being given relationships between identities, we infer them by observing ongoing interactions.

III. Our Approach

Our approach to entity resolution identifies relationships between identities and the roles that identities play in their social groups and then uses that information, along with the attributes of the identities to resolve identities to their underlying entities. Given a set of data that contains communications between identities over a period of time, our approach searches for the best mapping X of identities to entities that maximizes the following objective function

$$\mathsf{Fit}(X) = \alpha \cdot \mathsf{AttributeFit}(X) + \beta \cdot \mathsf{RelationshipFit}(X) + \gamma \cdot \mathsf{RoleFit}(X) \quad (1)$$

In this objective function,

- *Attribute Fit* captures that correctly resolved identities often have the same characteristics.
- *Relationship Fit* captures that correctly resolved identities often interact with the same entities.
- *Role Fit* captures that correctly resolved identities often have the same role and behavior patterns.

Consider the example of this objective function in the domain of social media. Attribute Fit captures that a Twitter profile may have the same name as a Facebook profile, which makes it more likely that they are the same person. Relationship Fit captures that a Twitter user may interact with her best friend on both Twitter and Facebook. Role Fit captures that a social maven is likely to be popular on both Twitter and Facebook.

To further illustrate the point, consider the example of a fraud group that is broken up by authorities but reforms under a different set of entities, and we seek to use this objective function to reidentify members. Attribute Fit captures that the new identity of a particular fraudster may share characteristics with his or her previous identity, such as zip code. Relationship Fit captures that the fraudster will still likely interact with the same entities as before, even though those entities may now be represented with new identities. Role Fit captures that the fraudster will still serve the same role as before, such as being a leader or a liaison between groups of fraudsters.

The optimization function $\mathsf{Fit}(X)$ gives a score to a mapping X of identities to entities. Since we are seeking to maximize $\mathsf{Fit}(X)$, and each term is additive, we add a penalty constant for each pair of identities mapped to an entity. We also take the transitive closure of pairs. If identities A and B are mapped to the same entity in a solution, and identities B and C are mapped to the same entity in the same solution, we also map A to C. In this section, we will first discuss how $\mathsf{Fit}(X)$ is implemented. We will then discuss how our algorithm searches for the mapping X that maximizes $\mathsf{Fit}(X)$ using a genetic algorithm.

A. Optimization Function Implementation

1) Attribute Fit: Attribute Fit measures how well the attributes overlap for all resolved pairs of identities. For each pair of resolved identities (x, y), Attribute Fit is computed as the attribute-specific distance $\mathsf{dist}_a(x, y)$ times an attribute-specific weight ϕ_a, summed over all attributes a.

$$\mathsf{AttributeFit}(x, y) = \sum_a \phi_a \mathsf{dist}_a(x, y) \quad (2)$$

The function $\mathsf{dist}_a(x, y)$ is specific to the attribute a. For names, the simplest implementation would be to use the edit distance, and other implementations are possible for other attributes. The weights ϕ_a can be learned if one has labeled training data using a perception or the Naive Bayes method.

2) Relationship Fit: We identify relationships in data using Temporal Group Detection [4]. Temporal group detection analyzes communications over time and finds groups of individuals who consistently communicate. Any identity x that is found to be in the same group as another identity y, and communicates directly with entity y at least once, is considered to be a neighbor of identity y. In this case, we consider there to be a relationship between identity x and identity y.

Relationship Fit measures how the resolved pairs of entities fit together with their neighboring entities. An example is shown in Fig. 2. The identities Bob T. and John M. are known to communicate, and the identities Rob T. and Jon M. are known to communicate. If the algorithm resolves Bob T. with Rob T., then Relationship Fit makes it more likely that John M. and Jon M. will be resolved.

Fig. 2: Collective entity resolution. If one determines that Bob T. and Rob T. are the same entity, and there are relationships between Bob T. and John M. and between Rob T. and John M., then one is more likely to recognize that Jon M. and John M. are the same entity.

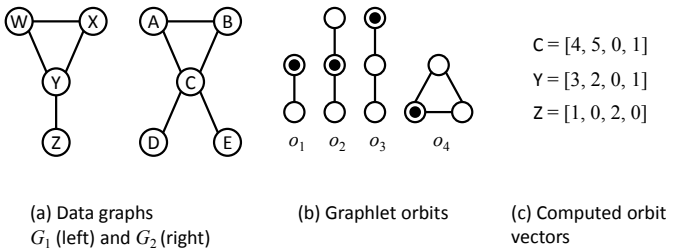


Fig. 3: A small graphlets example using four orbits. (a) Shows two graphs G_1 and G_2. We wish to compare the similarity of the roles of nodes in graphs G_1 and G_2. (b) Four orbits that serve as the features with which we will compare nodes from graphs G_1 with those of G_2. (c) The feature vectors for three nodes using the four graphlet orbits.

For each pair of resolved identities (x, y), the relationship fit is computed by looking at the resolved entities of the identities of their neighbors, call these sets of entities $N(x)$ and $N(y)$ respectively. To measure the entities in common between x and y, we use the Jaccard distance, which is the intersection divided by the union.

$$\mathsf{RelationshipFit}(x, y) = \frac{|N(x) \cap N(y)|}{|N(x) \cup N(y)|} \tag{3}$$

Relationship Fit is collective. This means that the resolution of the neighbors of an identity influences how that identity will be resolved [1], as shown in Fig. 2.

3) Role Fit: Role Fit measures how well the roles in their respective modalities overlap for resolved identities. Consider the fraud example in the introduction where the group is disbanded and sets up under a new set of identities. The leader of that group will still likely act as the leader, and in general, the members will still fulfill the same roles. We use Role Fit to help identify the members using the role they play. In contrast with Relational Fit, we do not use group detection to determine relationships. With Role Fit, we want to look at the behavior of the identities in the wider environment, not just within its group. To compute Role Fit, we look at the data graph, which consists of identities as nodes and observed interactions (such as communications) between identities as edges.

For each pair of resolved identities (x, y), the Role Fit is computed by comparing their roles in the data graph using graphlets. Graphlets can be used to compare the similarity of relationships between two networks [7]. Each graphlet is a small graph fragment that acts as a feature, roughly analogous to how wavelet-like features are used in computer vision tasks, such as face detection [8]. To characterize a particular node in a graph, the algorithm sees how many times each feature can be applied to that node. The graphlets often contain symmetries, so we only want to count how many times that node can be involved for each symmetry class, called an orbit. The result is a vector for each node that characterizes the node's structure.

Looking at Fig. 3, suppose we want to compare how similar node C in graph G_2 is compared to node Y in graph G_1. Four orbits are shown in Fig. 3(b), and using those four orbits, we can characterize node C with a vector of [4, 5, 0, 1]. The first value '4' is because C takes the place of orbit o_1 in G_2 four times: $\{\{C, D\}, \{C, E\}, \{C, A\}, \{C, B\}\}$. The second value '5' is because C takes the place of orbit o_2 in G_2 five times: $\{\{A, C, D\}, \{A, C, E\}, \{B, C, D\}, \{B, C, E\}, \{D, C, E\}\}$. Note that $\{A, C, B\}$ does not count because graphlets must match on the edges that are not there as well as the edges that are there. Because there is an edge $\{A, B\}$, the orbit o_2 does not match $\{A, B, C\}$.

Once we have computed the orbit vectors for two nodes, we can compare how similar those nodes are by looking at how similar their vectors are using Euclidean distance or cosine distance or a graphlet-specific distance metric. We see that nodes C and Y are similar, but when we compute the vector for node Z, we see that is substantially different than the vector for node C.

We use graphlets to compare pairs of nodes in Role Fit. The graphlets we use contain up to four nodes, which consist of 15 orbits. We, therefore, can represent the role of each node with a vector of size 15. These vectors are then compared using the graphlet specific distance measure of [9] to compute similarity. To compute the orbit vectors in a computationally efficient manner, we use a combinatorial approach [10].

B. Genetic Algorithm Search Implementation

We search for an optimal solution to the optimization function $\mathsf{Fit}(X)$ using a genetic algorithm. A genetic algorithm implementation consists of a set of chromosomes, a fitness function, a mutation operator, and a crossover operator. Each chromosome encodes a possible solution. At each iteration (generation), the mutation operator is applied to each chromosome, possibly creating a new potential solution. Pairs of chromosomes are also merged together into new potential solutions using the crossover operation. The crossover operation allows chromosomes to exchange information, and the goal of crossover is to combine large blocks of bits that have evolved independently, with the hope of speeding up search. The optimization function $\mathsf{Fit}(X)$ serves as the fitness function, and it is applied to individual chromosomes to determine which ones are the fittest and should survive to the next generation.

In our implementation, each chromosome in the genetic algorithm takes the form of a graph. The nodes and the graph are identities, and each edge in each graph indicates a pair of identities resolved to the same entity. We create an initial population of chromosomes, and then at each generation we randomly choose an evolutionary operator. Our evolutionary operators are mutation type I, mutation type II, and crossover.

1) Initial Population Created with GRASP: The initial population of chromosomes is created using a version of the Greedy Randomized Search Procedure (GRASP) [11]. GRASP

works by randomly generating a candidate solution and then performing a local search to see if it can be improved. This process of generating candidate solutions and performing local hillclimbing is performed multiple times, and the best solution is taken.

In our implementation of GRASP, we initially compute the sum of Attribute Fit and Role Fit[1] for each pair of identities, and we set aside those pairs that exceed a minimum threshold.[2] We then initialize each chromosome by randomly resolving pairs of identities in that set aside collection, with the probability of resolution being proportional to their sum of Attribute Fit and Role Fit.

2) Mutation Operator Type I: Based on Relationship: This type of mutation operation is based on the fact that entities that share the same neighbor(s) are more likely to be resolved. Based on resolved identities, we look for potential candidates among their neighbors to be resolved.

Similar to how clusters of resolved entities are merged in Bhattacharya and Getoor [1], we create clusters of identities that are resolved to the same entity using transitive closure. If identities x and y are resolved together, and identities y and z are resolved together, then $\{x, y, z\}$ form a cluster. The set of clusters is denoted by $\mathcal{C}$. Recall from the discussion of Relationship Fit that two identities are considered to be neighbors of one another if they are in the same temporal group, as found by temporal group detection, and there has been at least one direct communication observed between the identities. A cluster $c_2 \in \mathcal{C}$ is considered to be a neighbor of a cluster $c_1 \in \mathcal{C}$ if any identity in cluster c_1 is a neighbor an identity in cluster c_2.

For each pair of cluster neighbors $(c_1, c_2) \in \mathcal{C}$, we define the similarity $\mathsf{sim}_{(c_1 c_2)}$ between clusters as

$$\mathsf{sim}_{(c_1 c_2)} = \min_{(n_1 \in c_1, n_2 \in c_2)} \mathsf{AttributeFit}(n_1, n_2) \qquad (4)$$

If $\mathsf{sim}_{(c_1 c_2)}$ is more than a threshold μ, with probability of $\mathsf{sim}_{(c_1 c_2)}$, we add edges to the graph representation of the chromosome to merge those two clusters of identities into a single entity.

3) Mutation Type II: Large Neighborhood Search: Mutation of chromosomes is done using Large Neighborhood Search (LNS). Neighborhood search entails evaluating all of the possible solutions that are close (in the neighborhood) to the current solution to see if any of those solutions are better than the current one. Large Neighborhood Search entails considering a very large set of possible neighbors where those neighbors are too numerous to explicitly enumerate [12].

In our implementation, we make multiple changes to each chromosome at each generation. For each change, we flip a coin to determine if it will be an addition or a deletion. If it is an addition, we choose an edge $x = (x_1, x_2)$ to be added (among the pairs of identities not currently resolved by an edge) probabilistically based on the probability

$$p(x) = \frac{1}{\alpha}\mathsf{AttributeFit}(x_1, x_2) + \mathsf{RoleFit}(x_1, x_2) \qquad (5)$$

where is a normalizing constant. For deletions, we perform the analogous operation using

$$p(x) = 1 - \frac{1}{\alpha}\mathsf{AttributeFit}(x_1, x_2) + \mathsf{RoleFit}(x_1, x_2) \qquad (6)$$

4) Crossover Preserves Common Edges in Parents: The crossover operation intensifies the search around good solutions. In our implementation, it preserves common edges in parents while probabilistically adding edges (with probability 0.5) that reside in only one of the two chromosomes.

IV. Experimental Results

We perform two sets of exploratory experiments. The first set of experiments is on two standard datasets to establish that our algorithm is on par with existing ones. These datasets are bibliographic data, and it is not necessary to infer relationships. The second set of experiments is on a Twitter dataset that we collected where we look at the communications between Twitter users to infer relationships. We use the inferred relationships for Relationship Fit and the raw communications for Role Fit.

A. Standard Bibliographic Datasets

We used two datasets for this experiment. One is the CiteSeer dataset, which contains 1,504 machine learning papers with 2,892 author references to 1,165 author entities. The other is an arXiv dataset that contains 29,555 papers in high energy physics domain with 58,515 author references to 9,200 authors. For both datasets, the only attribute information available is the author's name. Each name contains the full last name and either full first name or the initials of the first name. Each author is an identity, and there is a relationship between author identities if they co-authored a paper.

Because there is no concept of behavior in these datasets, we only use Attribute Fit and Relationship Fit in these experiments. The similarity of authors names is a weighted sum of the similarities of their last names and first names. We use scaled edit distance to measure the similarity of the authors last names. For similarity measure of their first names, we take into consideration that full first names of the authors are not always available. When both full first names are available, we use scaled edit distance. Otherwise, we compare their initials. Two initials are determined to be similar only when one initial sequence is contained in another (e.g., "c. h." or "c. i." are sequences contained in "c. i. h.", but "h. c." is not contained in "c. i. h").

We perform the experiments with only Attribute Fit and with both Attribute Fit and Relationship Fit. We present the best result obtained in terms of F-measure in Table I. In this table, we also include the best results in the literature from Bhattacharya and Getoor [1]. Our results are comparable to the state of the art. We obtained better results than the best known result on the CiteSeer dataset, while our result on arXiv dataset is the same as the best known result.

B. Collected Twitter Dataset

We use Twitter data to evaluate the ability of our algorithm to perform entity resolution when the entities must be inferred.

[1]Relationship Fit is not available since at this point we have not made resolutions.

[2]With large sets of nodes, it becomes necessary to initially block them so that only likely candidates are pair-wise compared.

TABLE I: F-Measure of Experiment Results on Author Datasets

	CiteSeer	arXiv
Attribute Fit	0.982	0.980
Attribute Fit + Relationship Fit	0.996	0.985
Bhattacharya and Getoor (2007)	0.995	0.985

Twitter does provide lists of followers and friends, but many real-world use cases such as fraud or intelligence analysis do not come with lists of relationships, and these relationships must be inferred. Fraud and classified data can be difficult to use for scientific publications, so we use Twitter data but ignore the friend and follower list to simulate this restriction.

We collected two sets of data tweets from Austin, Texas. The first collection ran from October 31, 2013 to November 12, 2013. It contains 138,068 edges and 96,054 user nodes derived from 175,756 tweets. The second dataset was collected from November 13, 2013 until November 26, 2013. It contains 157,254 edges and 106,495 user nodes derived from 196,580 tweets. In the Twitter dataset, identities are Twitter users. Instead of using follower lists, we look at communications between Twitter users to infer relationships. We declare that a communication between two Twitter identities has occurred when one identity mentions another in a tweet. In Twitter, user screen names begin with the '@' symbol, so it is easy to determine who is being mentioned. There can also be multiple communications in a single tweet. For example, if @Henry tweets: "I love getting ice cream with @Monica and @Rita! We declare that one communication has taken place between @Henry and @Monica and another communication has taken place between @Henry and @Rita.

To focus our attention on well established networks among vast twitter users, we ran Temporal Group Detection [4] and found 22 groups in the first dataset and 25 groups in the second dataset. We treat these two datasets as if they were from different modalities and we seek to resolve members of groups across the two datasets. We used screen names (also known as user names, e.g., @Henry) as ground truth, but these screen names were not given to the algorithm (nor were the names, e.g., "Henry Johnson"). The attributes for each Twitter identity consisted of

- a TF/IDF vector of everything that user tweeted during that time period,
- the number of followers for that user,
- the number of people the person follows (Twitter calls these friends),
- the number of tweets that person has tweeted,
- a string representation of the user's location, inputted by the user. In this dataset, most strings are some derivation of "Austin, Texas."

There were 196 members in the 22 groups in the first dataset and 214 members in the groups in the second dataset, resulting in 41,944 possible resolutions.

TABLE II: Best F-Measure Achieved in Each Condition over a Range of Penalty Values on Twitter Dataset

	Pairwise-Group	Single-Group	Pairwise-Comm	Single Comm
Attribute only	0.853	0.873	0.853	0.852
Attribute Fit + Role Fit	0.853	0.893	0.853	0.873
Attribute Fit + Relationship Fit	0.877	0.893	0.877	0.893
Attribute + Role + Relationship	0.877	0.893	0.877	0.893

1) Experimental Conditions: To evaluate the contribution of Relationship Fit and Role Fit to solution quality, we perform experiments under the following settings: Attribute Fit only; Attribute Fit and Role Fit; Attribute Fit and Relationship fit; and Attribute Fit, Relationship Fit, and Role Fit. These are evaluated under four conditions stemming from two choices.

The first choice is how many times to run entity resolution.

- *Pairwise* means to perform entity resolution multiple times, once for each pair of groups. Recall that we found 22 groups in the first dataset and 25 groups in the second dataset. This condition runs our entity resolution algorithm $22 \times 25 = 550$ times, once for each pair of groups. For each pair, it counts the number of true positives, false positives, true negatives, and false negatives. After all runs, these are summed.
- *Single* is the alternative to Pairwise. Single means to run entity resolution once for all the groups. For the Single condition, we run entity resolution once and count the number of true positives, false positives, true negatives, and false negatives.

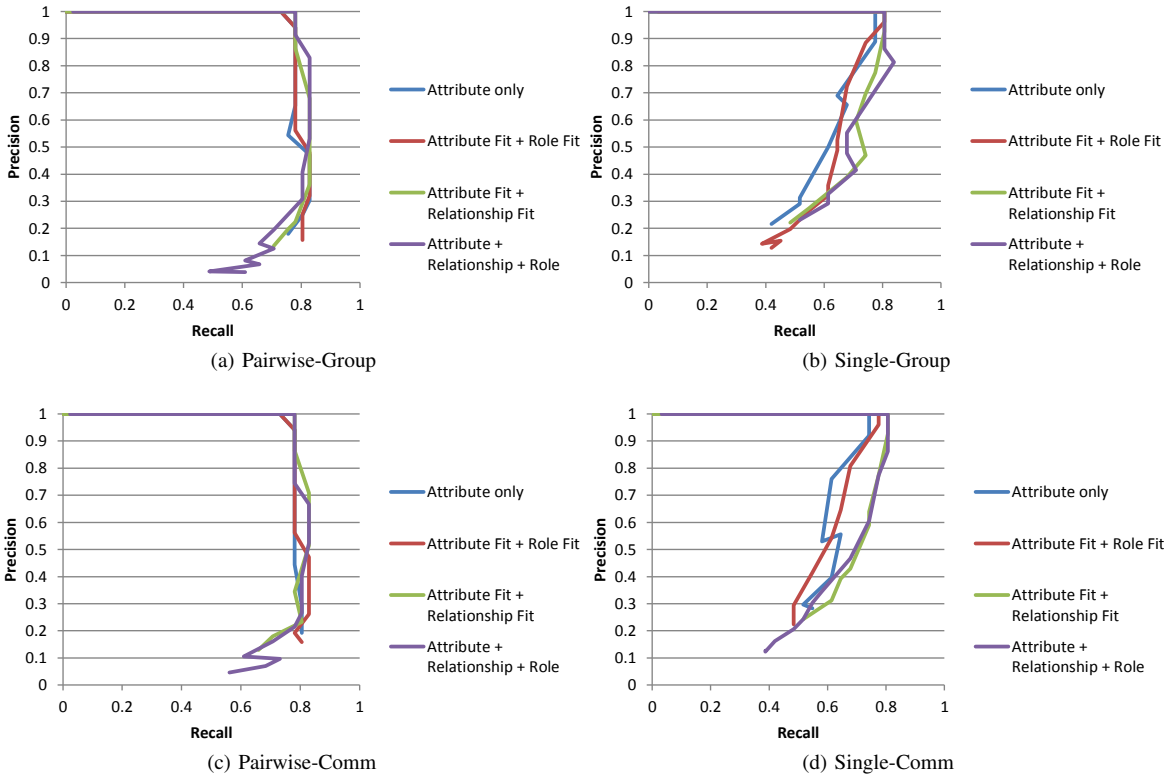
The second choice is how to define Role Fit.

- *Comm* means to implement Role Fit how it was described in the body of the paper using graphlets, where the graph is the underlying data graph consisting of all communications.
- *Group* is the alternative to Comm. It calculates Role Fit based on how we define relationships, which requires that edges in graphlets be communications between two identities in the same group.

In total, there are four conditions: Pairwise-Comm, Pairwise-Group, Single-Comm, Single-Group.

2) Experimental Results: The accuracy is usually well over 99% because there are so many true negatives. We therefore look at the F-Measure. We ran each condition over a range of penalty values, and Table II shows the best F-Measure achieved in each condition. The precision-recall curves over the weight of the penalty function are shown in Fig. 4.

Relationship Fit and Role Fit allow one to obtain better results. The results indicate that graphlets (Role Fit) can be a stand-in when relationships cannot be inferred. The precision recall curves show a curious behavior of curving back in as the penalty function loosens. This is presumably because the loose penalty function causes the algorithm to make incorrect resolutions early in the search process that prevent it from finding correct resolutions later in the search process.



(a) Pairwise-Group (b) Single-Group

(c) Pairwise-Comm (d) Single-Comm

Fig. 4: Precision-Recall curves in each condition over a range of penalty values on the Twitter dataset. These graphs offer a different perspective on how Relationship Fit and Role Fit improve performance.

V. Conclusion

As our ability to collect data about the world has expanded, entity resolution has become increasingly important for combating fraud and for understanding complex environments with multiple data sources. In this work, we focus on the task of entity resolution in relational environments. In relational environments, besides having attributes, entities relate to other entities, and those relations can be used to help resolve entities.

Our exploratory experiments indicated that we can identify relationships by observing behavior and that these relationships can aid in entity resolution. Relationships are found using a temporal group detection algorithm applied to observed communications between identities. The experiments also provided evidence that graphlets, small graphs that act as features to describe nodes in a graph, are helpful for entity resolution.

Acknowledgment

The authors would like to thank ONR for sponsorship of this effort.

References

[1] I. Bhattacharya and L. Getoor, "Collective entity resolution in relational data," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 1, no. 1, p. 5, 2007.

[2] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Security and Privacy, 2009 30th IEEE Symposium on*. IEEE, 2009, pp. 173–187.

[3] O. Peled, M. Fire, L. Rokach, and Y. Elovici, "Entity matching in online social networks," in *Social Computing (SocialCom), 2013 International Conference on*. IEEE, 2013, pp. 339–344.

[4] J. Mugan, E. McDermid, A. McGrew, and L. Hitt, "Identifying groups of interest through temporal analysis and event response monitoring," in *IEEE Conference on Intelligence and Security Informatics (ISI)*, 2013.

[5] P. Singla and P. Domingos, "Entity resolution with Markov logic," in *Sixth International Conference on Data Mining (ICDM'06)*. IEEE, 2006, pp. 572–582.

[6] S. Bartunov, A. Korshunov, S. Park, W. Ryu, and H. Lee, "Joint link-attribute user identity resolution in online social networks," in *The Sixth SNA-KDD Workshop Social Network Mining and Analysis*, 2012.

[7] N. Pržulj, "Biological network comparison using graphlet degree distribution," *Bioinformatics*, vol. 23, no. 2, pp. e177–e183, 2007.

[8] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[9] T. Milenković and N. Pržulj, "Uncovering biological network function via graphlet degree signatures." *Cancer informatics*, no. 6, 2008.

[10] T. Hočevar and J. Demšar, "A combinatorial approach to graphlet counting," *Bioinformatics*, p. 717, 2013.

[11] M. G. Resende and C. C. Ribeiro, "Greedy randomized adaptive search procedures: Advances, hybridizations, and applications," in *Handbook of Metaheuristics*. Springer, 2010, pp. 283–319. [Online]. Available: http://www2.research.att.com/ mgcr/doc/sgrasp-hmetah.pdf

[12] R. K. Ahuja, J. B. Orlin, and D. Sharma, "Very large-scale neighborhood search," *International Transactions in Operational Research*, vol. 7, no. 4-5, pp. 301–317, 2000.

[13] T. Coffman, "A method and apparatus for fusion of multi-modal intelligence data," USA Patent Provisional Application No. 61/506,582, 2011.