Improving Entity Resolution with Global Constraints

[Microsoft Research Technical Report MSR-TR-2011-100]

Jim Gemmell Benjamin I. P. Rubinstein Ashok K. Chandra
Microsoft Research, Mountain View, CA
{jgemmell, ben.rubinstein, achandra}@microsoft.com

ABSTRACT

Some of the greatest advances in web search have come from leveraging socio-economic properties of online user behavior. Past advances include PageRank, anchor text, hubs-authorities, and TF-IDF. In this paper, we investigate another socio-economic property that, to our knowledge, has not yet been exploited: sites that create lists of entities, such as IMDB and Netflix, have an incentive to avoid gratuitous duplicates. We leverage this property to resolve entities across the different web sites, and find that we can obtain substantial improvements in resolution accuracy. This improvement in accuracy also translates into robustness, which often reduces the amount of training data that must be labeled for comparing entities across many sites. Furthermore, the technique provides robustness when resolving sites that have some duplicates, even without first removing these duplicates. We present algorithms with very strong precision and recall, and show that max weight matching, while appearing to be a natural choice turns out to have poor performance in some situations. The presented techniques are now being used in the back-end entity resolution system at a major Internet search engine.

Categories and Subject Descriptors

H.2 [**Information Systems**]: Database Management; H.3.3 [**Information Systems**]: Information Search and Retrieval; I.5.4 [**Pattern Recognition**]: Applications

General Terms

Algorithms, Experimentation

Keywords

Entity resolution, semantic web

.

1. INTRODUCTION

Some the greatest advancements in web search have come from leveraging properties of the web that arise due to social behavior or economic interests. PageRank uses one such *socio-economic property* of the web that reflects the preference of webmasters to link to popular pages, and search engines use PageRank to figure out the most relevant pages for any query [18]. Anchor text is another socio-economic property that represents the keywords by which the rest of the world refers to a given page, even if these keywords are not present in the page itself. By exploiting these two socio-economic properties together, Google made great advances over previous search engines. Kleinberg [14] utilized another socio-economic property of the web: that for various topics there are people who make lists of good web pages, and the bipartite graph of these hubs (list pages) and authorities (linked pages) are mutually reinforcing.

In this paper we investigate another online socio-economic property that, to our knowledge, has never been exploited: that sites listing entities have an incentive to avoid gratuitous duplicates. For instance, duplicate movies in IMDB would have reviews and corrections applied to one copy and not the other. If Netflix has one entry for a DVD and a duplicate for the Blu-ray version, then their customers might be looking at one and not realize the other is available. Hulu supports Facebook likes for their movies and could have the like counts diluted by duplicates. Additional examples in other domains are easily constructed. We leverage this socio-economic property to resolve entities across the different web sites, by applying a global one-to-one constraint to produced matchings. The resulting resolution has much better accuracy compared to matching without such a constraint. Our framework for one-to-one entity resolution (ER) is generic in that it can constrain existing resolution methods using weighted graph matching. Our goal is not to engineer features or tune high-performance domain-specific ER, but rather to develop generic algorithms that can be combined with existing methods for improving retrieval performance.

We perform a large-scale case-study in resolving movie entities from the complete catalogs of IMDB, Netix, iTunes and AMG, the largest of which contains roughly 500k entities. We use an active learning-based logistic regression score combiner with simple features and blocking, and find that constrained ER significantly outperforms unconstrained resolution, while being resilient to poor score tuning. When compared with a crowd-sourced dataset of 13K movie matches from Freebase, our algorithm improves on the existing high precision, and increases recall by a factor of over three.

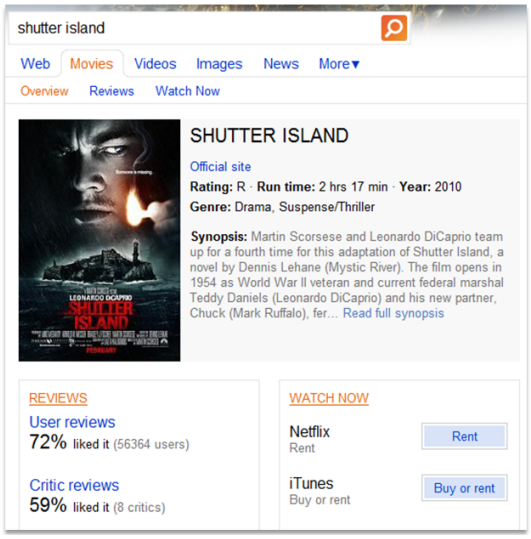

Figure 1: Entity resolution is used in a major search engine's movies vertical for implementing "buy", "rent", "watch" entity actions and surfacing reviews.

The system presented here forms the basis for entity resolution as used in production in one of the major search engines on the Internet today. The system aggregates movie information from a number of sources in order to support entity actions like "rent", "watch", and "buy", as well as aggregating review information (*cf.* Figure 1). Because many sources need to be merged, we aim to minimize the labeling and training needed to incorporate new sources, in addition to requiring excellent precision and recall. It turns out that the one-to-one matching algorithms presented here are surprisingly robust, in that a scoring function for resolving entities across two sites often works effectively for matching other sites as well.

Finally we present a surprising result that demonstrates that the problem of graph matching for constrained entity resolution is fundamentally new—specifically that maximizing match weight does not necessarily optimize precision and recall—opening up a new direction for database and graph algorithm research. While exact max weight matching generally performs very well, it suffers extremely poor performance when very similar entities exist, resulting in both false negatives and false positives.

1.1 Related Work

Numerous previous works have studied entity resolution [28, 22, 15, 2, 16], statistical record linkage [12, 26, 7, 27] and deduplication [21, 3, 9, 11]. There has been almost no previous work producing one-to-one resolutions, and to our knowledge our work is the first to study the benefits of leveraging one-to-one structure in resolution problems, and the first to recognize the novel problem of graph matching for entity resolution.

Some works have looked at constraints in general [5, 23, 1] but they do not focus on the one-to-one constraint. Su *et al.*, do rely on the rarity of duplicates in a given web source [24] but use it only to generate negative examples. Jaro's work on linking census data matches records using a Linear Program formulation that enforces one-to-one resolution [13]. However no justification or evaluation of the constraint is offered, and the LP formulation can be unnecessarily expensive. The focus of this paper sits squarely on evaluating the benefits of globally constraining entity resolution. We show that while our best-performing one-to-one ER algorithms incur only slightly more computational cost than the unconstrained approach, their statistical performance is significantly greater. We also shed light on the problem of graph matching to maximize precision/recall, demonstrating that it does not simply reduce to max weight graph matching.

Many prior studies have explored resolving entities with mixed-type attributes, by comparing individual attributes to get feature-level scores and then using machine learning to combine these scores into an overall match score [15, 28, 16]. A significant number of these studies have considered sophisticated feature-level scoring techniques particularly for strings [8] and have compared several standard machine learning combiners including SVMs [21, 3, 15, 28], decision trees [21, 25, 3, 15, 28], and Naïve Bayes [21, 28]. Our goal in this paper is to develop and analyze effective one-to-one resolution algorithms that are generic in nature and do not rely on feature-level scoring or scoring combiners that are highly tuned using human domain expertise. Thus while our experimental comparison of resolving movie entities involves mixed-type entity attributes, we chose to use very simple feature-level comparisons and a logistic regression combiner that is has been used in entity resolution previously [22, 19, 28, 15, 6]. Even with an arguably simple approach to scoring we show that entity resolution can efficiently be constrained to produce one-to-one matches.

2. GENERIC ENTITY RESOLUTION

The purpose of this paper is not to investigate alternate scoring functions, but rather to explore generic algorithms for constrained ER. In this section, we describe the abstract ER problem, the framework for including particular scoring approaches, and several generic algorithms for constrained ER. After describing the generic ER algorithms in this section, we provide specific details of our basic scoring for movie entities in Section 3.

We begin with notation of the abstract ER problem. Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two *databases* each containing a finite number of *rows* representing *entities*. The dataset sizes need not be the same, and each dataset represents its entities via an encoding in some set $\mathcal{X}_i$. The goal of the *Entity Resolution Problem* is to identify an unknown target relation or *resolution* $\mathcal{R} \subseteq \mathcal{D}_1 \times \mathcal{D}_2$, given some information about $\mathcal{R}$; the quality of a retrieved relation $\hat{\mathcal{R}} \subseteq \mathcal{D}_1 \times \mathcal{D}_2$ is measured by precision and recall. In the exact one-to-one case, the target *matching* $\mathcal{R}$ is one-to-one: for each $x_1 \in \mathcal{D}_1$ there exists at most one $y \in \mathcal{D}_2$ such that $(x_1, y) \in \mathcal{R}$, and similarly for each $y_2 \in \mathcal{D}_2$ there exists at most one $x \in \mathcal{D}_1$ such that $(x, y_2) \in \mathcal{R}$.

It is common-place in many-to-many ER to leverage sets of known pairs from $\mathcal{R}$ and its complement, to learn a scoring function $f : \mathcal{X}_1 \times \mathcal{X}_2 \to \mathbb{R}$ such that $\hat{\mathcal{R}}$ is taken to be all pairs in $\mathcal{D}_1 \times \mathcal{D}_2$ scoring above a threshold $\theta \in \mathbb{R}$ [3, 6, 15, 16, 19, 21, 22, 25, 28].

We presume an abstract scoring process composed of several stages: First, to improve runtime performance, we block pairs so that not all pairs in the Cartesian product are scored. After blocking, feature-level scores are generated, which are then combined into an overall score using a su-

pervised learner (logistic regression here). Active learning is employed to reduce the required number of human-labeled training examples. This results in a many-to-many resolution; see Appendix A for more details on this abstract scoring process.

We now describe five generic algorithms for constrained ER. The first two, ManyMany and FirstChoice are not one-to-one, but rather serve as baseline comparisons. The remaining three algorithms apply one-to-one constraints.

2.1 Unconstrained ManyMany

For unconstrained entity resolution we employ the popular approach [3, 6, 15, 16, 19, 21, 22, 25, 28] described above which we call ManyMany: given a score function $f : \mathcal{X}_1 \times \mathcal{X}_2 \to \mathbb{R}$ and threshold $\theta \in \mathbb{R}$ simply resolve all tuples with score exceeding the threshold. The statistical performance of ManyMany relies on score function $f(\cdot)$ to rank pairs in order of likelihood of belonging to $\mathcal{R}$. We shall see that the remaining resolution algorithms exploit structure of $\mathcal{R}$ to weaken their reliance on a robust $f(\cdot)$. As is common in machine learning and information retrieval, decreasing the threshold θ trades-off precision for recall. ManyMany takes $\mathcal{O}(|\mathcal{D}_1| \cdot |\mathcal{D}_2|)$ evaluations of $f(\cdot)$, and the same space.

2.2 One-to-Many FirstChoice

As a first step towards globally constrained one-to-one ER, we may consider constraining ManyMany to being one-to-many (or many-to-one). The simplest method of achieving this constraint is to resolve $x_1 \in \mathcal{D}_1$ with a row of $\mathcal{D}_2$ achieving the highest score with x_1, provided that this score exceeds threshold θ; if no incident edge is scored higher than θ then x_1 goes unmatched:

$$\hat{\mathcal{R}}_{\text{1-M}} = \{(x_1, x_2) \in \mathcal{D}_1 \times \mathcal{D}_2 | \forall x \in \mathcal{D}_2, f(x_1, x_2) \geq \theta, f(x_1, x)\}.$$

Similarly for many-to-one $\hat{\mathcal{R}}_{\text{M-1}}$. This algorithm, which we name FirstChoice, requires the same time complexity as ManyMany, and a reduced $\mathcal{O}(|\mathcal{D}_1|)$ space complexity.

2.3 One-to-One Entity Resolution

The added structure of the one-to-one resolution problem suggests that the task be viewed in terms of graph matching. The edge set $\mathcal{D}_1 \times \mathcal{D}_2$, together with the vertices $\mathcal{D}_1 \cup \mathcal{D}_2$ form a complete bipartite graph, weighted by the scores of edges. Target $\mathcal{R}$ now corresponds to a subset of edges that forms a matching in the graph theoretic sense. Clearly, a resolution $\hat{\mathcal{R}}$ that attempts to approximate $\mathcal{R}$ should be constrained to be one-to-one, which is in stark contrast with existing approaches in the entity resolution literature. Assuming that scores *well approximate* the likelihood of a match, retrieving $\mathcal{R}$ can be seen as selecting edges of high weight.

MutualFirstChoice. Our first one-to-one meta-algorithm, MutualFirstChoice, is simple and fast to run. It matches two entities iff they prefer each other at least as much as any other entity, with ties broken arbitrarily. This corresponds to running FirstChoice from each direction and combining the pairs in agreement, *i.e.*, taking $\hat{\mathcal{R}}_{\text{1-M}} \cap \hat{\mathcal{R}}_{\text{M-1}}$. This one-to-one resolution is relatively conservative, and should produce strong results on sparse pair scores.

Greedy. Our second simple one-to-one algorithm is Greedy. The meta-algorithm first sorts the tuples by decreasing score, discarding those falling below θ. It then steps through the remaining tuples in order: when a tuple is encountered, where neither of the involved entities have been previously matched, the tuple is included in the resolution. The runtime of Greedy increases by a logarithmic factor in the number of pairs for sorting. The algorithm's space complexity is $\mathcal{O}(|\mathcal{D}_1| \cdot |\mathcal{D}_2|)$, while storage of the resolution requires only $\mathcal{O}(\min\{|\mathcal{D}_1|, |\mathcal{D}_2|\})$ space.

MaxWeight. Our third generic one-to-one resolution algorithm, MaxWeight, uses exact max-weight bipartite graph matching. The Hungarian Method or augmenting paths are standard approaches to exact matching, and many approximations exist: indeed Greedy is a 2-approximation (although in our experiments it achieves significantly better approximations). The running-time of MaxWeight is cubic in the maximum database size, and space is the same as Greedy.

We view the constrained entity resolution problem as one of graph matching. Intuitively it seems natural to assume that MaxWeight should achieve the best performance since it selects pairs with the highest total score. However as we observe in Section 4 and discuss in Section 5 this is not always the case: the target objective of optimizing precision/recall does not generally correspond to maximizing weight.

3. EXPERIMENTAL METHODOLOGY

We implemented the five ER algorithms and applied them to the movies domain. The main sources we used for our experiments were the Internet Movie Database (IMDB)—often used as a primary source of metadata and reviews of movies on the Internet—and Netflix, a popular online rental and streaming service.

We also obtained movie data, both from crawling and querying APIs, from a number of other providers, including iTunes, AMG and Freebase. We focus our discussion mostly on IMDB and Netflix due to their popularity and large volume of data, while limiting details to proof of concept results on the other sources due to space limitations. In particular we use iTunes and AMG to demonstrate the resilience of constrained resolution using a model trained with IMDB and Netflix only. We also show that our approaches enjoy a large improvement to recall over the crowd-sourced movie entity resolution at Freebase.

3.1 Movie Datasets

We accessed the public APIs of both IMDB and Netflix to capture information about as many movies as possible (not including adult titles). Both data sources catalog a large number of movies: over 500k movies from IMDB and over 75k movies from Netflix. Both websites surface unique IDs for their movies, making it easy to distinguish movies within each site.[1]

IMDB generally has more metadata than Netflix, with more alternative titles, alternative runtimes, and a more comprehensive cast list. The Netflix catalog is much smaller than IMDB, as it is focused on movies that are popular as DVD rentals in North America. IMDB, on the other hand has has a much more extensive catalog of foreign movies, those unavailable on DVD, obscure titles, and so forth. How-

[1]For sites without a visible ID, we used the crawled URL. Conceivably the same movie can appear under different URLs, requiring de-duplication; but in the sites used, either the URL was unique or we could find an ID.

	Non-unique title	Non-unique title & date
Netflix	6%	0.4%
Imdb	19%	2%

Figure 2: Replicated movie titles make resolution on title alone infeasible. Surprisingly a non-trivial proportion of distinct movies released in the same year have identical titles.

ever, Netflix is not a subset of IMDB. There are thousands of movies on Netflix that cannot be found on IMDB.

Every movie from the sites we investigated had a title. However, other attributes were not universal.

As we explain below, the same title can appear in different forms, but it is also worth noting that titles are far from unique, justifying the use of additional features. We can find at least ten movies called "Avatar". Even in the same year, we observed multiple identically-titled movies (perhaps a form of movie spam). For example, there are several movies "Journey to the Center of the Earth" from 2008. Figure 2 summarizes statistics on replicated titles, and replicated titles with identical release years.

3.1.1 Human-Labeled Truth Sets

In order to train and evaluate algorithms for resolving movies, we gathered several collections of human-labeled truth sets.

- **IMDB Top 250.** IMDB maintains a list of the top 250 movies of all time ranked by user-submitted reviews [10]. For each entry a human used the Netflix domain search engine and Web search to find an appropriate Netflix match, which was possible for each movie.
- **Netflix Top 100.** Like IMDB, Netflix maintains a list of the 100 most popular movies among its subscribers. We followed the same procedure to find 100 known match pairs.
- **Random 350.** We selected a uniformly random sample of 350 movies from Netflix, and determined 233 matches within IMDB, noting those 117 movies that had no match.
- **Freebase** We used the Freebase API—an open repository of structured data—to find 13,005 movie entities linked to entities in both IMDB and Netflix. Due to the open crowd-sourced nature of Freebase we conjectured that the data would be mostly truthful, while suffering from some benign and perhaps malicious errors.
- **Boundary Dataset.** In order to train a machine learning-based score combiner, we collected a set of matches and non-matches that would be difficult to automatically identify. We begun with an initial human-tuned linear scoring function based on the feature-level scores described below. We then took a random sample of 1,234 IMDB-Netflix pairs whose scores were close to 0.5, and manually labeled these. 62% of the pairs were matches. This approach of training set selection corresponds to active learning [21].

3.2 Performance Metrics

We measure the performance of the five entity resolution algorithms via precision and recall. In order to evaluate these metrics, we run each algorithm on the entire IMDB-Netflix blocked set of pairs, and observe non/matched in the resulting $\hat{\mathcal{R}}$ compared to the aforementioned truth sets. Let $\mathcal{R}^\star_+$ and $\mathcal{R}^\star_-$ denote pairs in the truth set $\mathcal{R}^\star_+ \cup \mathcal{R}^\star_-$ known to be matches in $\mathcal{R}$ and pairs known not lie in $\mathcal{R}$ respectively. We measure the following counts

$$\begin{aligned} TP &= |\hat{\mathcal{R}} \cap \mathcal{R}^\star_+| \\ FN &= |\mathcal{R}^\star_+ \backslash \hat{\mathcal{R}}| \\ FP &= |\hat{\mathcal{R}} \cap \mathcal{R}^\star_-| \\ &\quad + \left|\left\{ (x,y) \in \hat{\mathcal{R}} \,\middle|\, \exists z \neq y, (x,z) \in \mathcal{R}^\star_+ \right\}\right| \\ &\quad + \left|\left\{ (x,y) \in \hat{\mathcal{R}} \,\middle|\, \exists z \neq x, (z,y) \in \mathcal{R}^\star_+ \right\}\right| \ . \end{aligned}$$

From these *precision* corresponds to $TP/(TP+FP)$ while *recall* is $TP/(TP+FN)$. Notice that false positives of the second and third kinds are inferred from the truth set. *Due to the movie data being one-to-one, a unique feature of our evaluation methodology is that we infer truth about pairs not explicitly in our truth set.*

3.3 Feature-Level Scoring

We retrieved the following attributes to use as features for each movie where available: titles, release year, runtimes, directors and cast. Space does not permit a detailed description of our feature-level scoring, but our approach is intentionally simple. In brief, our feature scores are:

- Title: Exact match yields a perfect score. Otherwise, the fraction of normalized words that are in common (slightly discounted). The best score among all available titles is used.
- Release year: the absolute difference in release years up to a maximum of 30.
- Runtime: the absolute difference in runtime, up to a maximum of 60 minutes.
- Cast: a count of the number of matching cast members up to a maximum of five. Matching only on part of a name yields a fractional score.
- Directors: names are compared, like cast. However, the number matching is divided by the length of the shorter director list.

Although the feature-level scores could be improved—*e.g.*, by weighing title words by TF-IDF, by performing inexact words matches, by understanding common title conventions like <title>:<subtitle>, or that the omission of "part 4" may not matter while the omission of "bonus material" is significant—our goal is to show that using the one-to-one constraint alleviates the need to be as elaborate in the development of feature scores. Our approach is intentionally minimalist.

3.4 Blocking

In order to avoid scoring all possible pairs of movies—which can grow to a very large number, almost 40b for IMDB-Netflix—we employ the standard technique of blocking (see *e.g.*, [16]). A key is used to define blocks of movie

Cast	*Title*	*Year*	*Directors*	*Runtime*	*Intercept*
1.56	1.13	-0.86	0.62	-0.31	0.82

Figure 3: The parameters of the logistic regression model, ordered by decreasing absolute value, fit to the Boundary dataset's training part.

pairs that are scored. In this case we score pairs whose titles share a normalized non-stopword, thereby producing overlapping blocks. We create an index of normalized non-stopwords for all movies. To deal with titles that contain no normalized words at all (such as the movie "+/-"), or only stopwords (such as the movies "At" and "3"), we insert special tokens in place of a non-stopword (EMPTY_NORMALIZED_TITLE and STOPWORDS_ONLY). We only score movies that have an entry in common in the index. Those that do not are presumed to have a low score, which we estimate as zero.

3.5 Learning How to Combine Scores

We used the core function for fitting generalized linear models in the R statistical computing environment [20] for training logistic regression. Before training the model, we first randomly partitioned the boundary dataset into a stratified partition of 856 training and 418 test examples. Each example consisted of the feature-level scores for title, year, runtime, director and cast matches (as described above), and a human label for whether the example pair is truly a match or not. We employed a simple active learning variant of logistic regression: as mentioned above, the boundary dataset was itself generated from examples close to an initial linear model's decision boundary.

The parameter vector resulting from centering, scaling then logistic regression learning is displayed in Figure 3. Note that the weights' signs are sensible: small year and runtime *differences* contribute to greater overall score, and are the only negative weights learned. By scaling and centering the data prior to learning, we may compare the relative contributions of the features by comparing the weights' relative magnitudes. Relative to runtime, which is the least predictive feature-level score, the cast's weight is over 5 times larger while the title and year weights are over 3.6 and 2.7 times larger respectively. While it may appear surprising at first glance that cast would be more predictive than movie title, it is clear that the cast is discriminative—even the most prolific stars appear in a limited number of movies, and the combination of stars makes the cast even more unique. This relative weight may also be an artifact of our training set, which included a number of non-matching movies with very high title scores. In terms of the model's statistical performance, logistic regression achieved a moderate test set accuracy of 89% under a threshold of 0.5.

4. RESULTS

This section presents our experimental results on comparing the five meta-algorithms for entity resolution using a thorough case-study on resolving movies entities, for the movie vertical of a major search engine.

4.1 Validating the One-to-One Assumption

To assess how close the true IMDB-Netflix matching is to being truly one-to-one, we scored all pairs of movies taken from within each dataset individually. We employed blocking, as two movies with no title word in common are unlikely to be duplicates. We then sorted the results by descending score, and sampled scored pairs at various intervals to get an estimate of the rate of duplicates, which we observed to be declining as the scores decreased, as expected. Based on this observed rate, we conservatively estimate that the true number of duplicates in each source is less than 500. We thus conclude that the IMDB-Netflix movie matching problem is in reality very near one-to-one. The duplicates that do exist are few in number and generally do not get matched to anything, so their impact on our results is not measurable.

4.2 Comparison of Meta-Algorithms

We evaluated the unconstrained, one-to-many and one-to-one entity resolution algorithms of Section 2, as described in Section 3. Figures 4–7 display Precision-Recall curves for all five algorithms, on each of the four truth sets from the IMDB Top 250 list, Netflix Top 100 list, a random sample of 350 Netflix movies manually matched with IMDB, and the boundary dataset's test sample.

Viewing the IMDB and Netflix lists of most popular movies as a surrogate for the head of Web or domain-specific movie search queries, we can regard Figures 4 and 5 as estimating the performance of the respective entity resolution algorithms on head movies. While the two one-to-one algorithms MutualFirstChoice and Greedy perform almost ideally on both datasets, ManyMany performs almost ideally only on the Netflix Top 100, where movies' Netflix and IMDB representations are complete and therefore easily matched, and fairs worse on the IMDB Top 250. Surprisingly MaxWeight performs as expected for only high thresholds receiving strong precision and recall, however the curve doubles back on itself in both figures for decreasing threshold. As we discuss in Section 5, unlike the other one-to-many and one-to-one meta-algorithms for which a decreasing threshold cannot affect previously resolved entities,[2] MaxWeight's matching can change drastically even for small decrements to the threshold, potentially producing subtly inappropriate resolutions.

Where the first two figures measure performance on the head, the curves induced by the Random 350 dataset shown in Figure 6 allow us to compare the meta-algorithms on random movies approximating tail performance. This evaluation tells a far different story: once again the unconstrained ManyMany performs the worst overall, however the one-to-many FirstChoice is now significantly worse than the one-to-one matchings whose performance differences are statistical insignificant. Notably MaxWeight's curve behaves like those of the other algorithms; as discussed in Section 5 this is due to different sampling used in this third evaluation set.

Figure 7 records the performance of the five resolution algorithms on the difficult Boundary dataset's independent test sample. This truth set is made up of examples close to the decision boundary of an initial linear classifier used in the active learning of logistic regression. Thus while we expect this set's examples to be very difficult to match correctly, we see moderate performance from the one-to-one

[2]For such algorithms, Precision-Recall curves should be roughly monotonic.

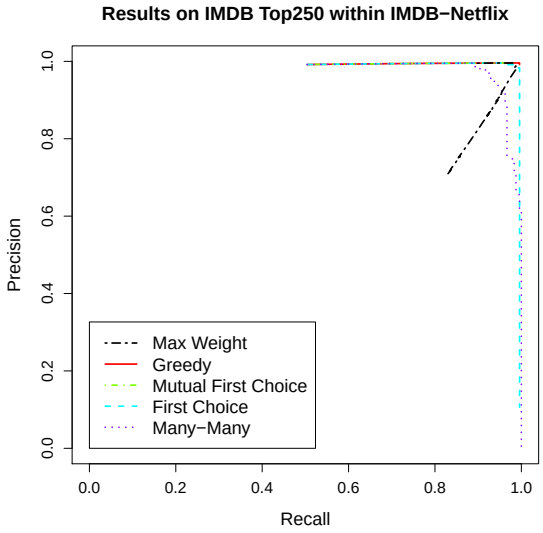

Figure 4: Results of testing the entity resolution algorithms on the IMDB Top 250 truth set.

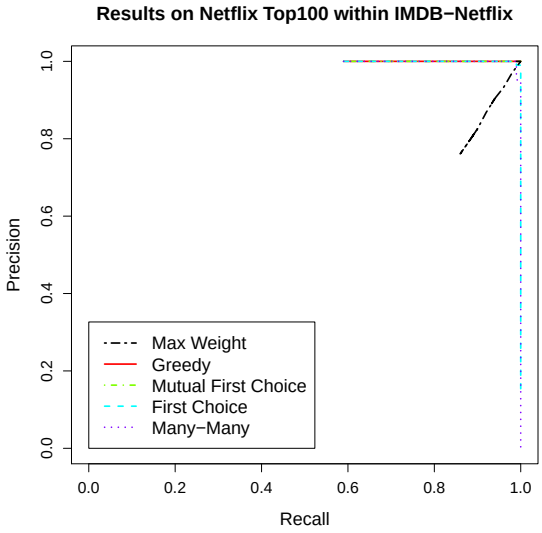

Figure 5: Results of testing the entity resolution algorithms on the Netflix Top 100 truth set.

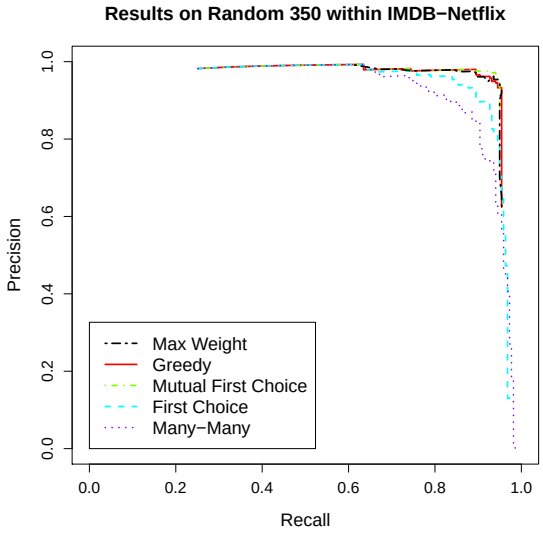

Figure 6: Results of testing the entity resolution algorithms on the Random 350 truth set.

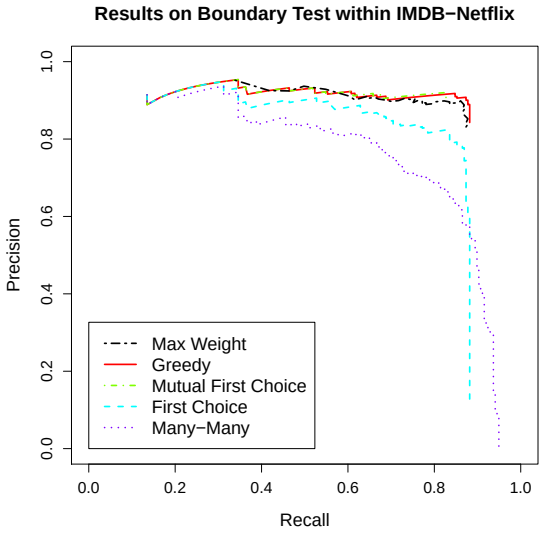

Figure 7: Results of testing the entity resolution algorithms on the Boundary dataset's test truth set.

meta-algorithms achieving both precision and recall close to 90% at the knee, compared to much poorer performances by FirstChoice and ManyMany with knee precision and recalls close to 83%, 81% and 81%, 69% respectively. As in the random set, MaxWeight has a curve that behaves like the other one-to-one algorithms.

From these results we conclude that *globally constrained resolution uniformly improves upon unconstrained resolution for the presented large-scale one-to-one movie resolution problem;* that *constrained one-to-one ER is significantly more effective than one-to-many ER;* and that *maximizing matching weight does not necessarily optimize performance of one-to-one ER.*

4.3 Crowd-Sourced Comparison

We compared our one-to-one resolution with a third party crowd-sourced entity resolution. Freebase is an open repository of structured data that contains many movie entities. We used the Freebase API to download movies and found 13,005 movies that linked to both Netflix and IMDB with IDs contained in our datasets. We compared the matches indicated by Freebase with our constrained resolution Greedy, using a threshold of 0.45 which roughly corresponded to the knee's of the Precision-Recall curves above. We found that out of the 13,005 pairs, our algorithm agreed with 12,695 (98%). We had a human evaluate the 310 additional pairs matched by Freebase and found that 156 (50%) were incorrectly matched. However, Greedy matched an additional

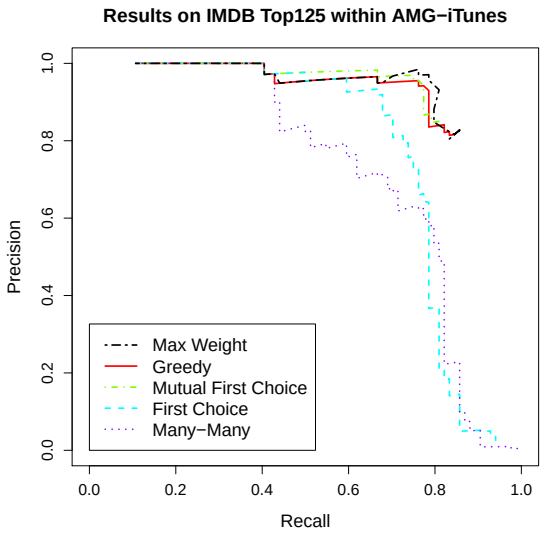

Figure 8: Results of testing the entity matching algorithms using IMDB-Netflix trained scores, on a truth set of AMG-iTunes popular movies.

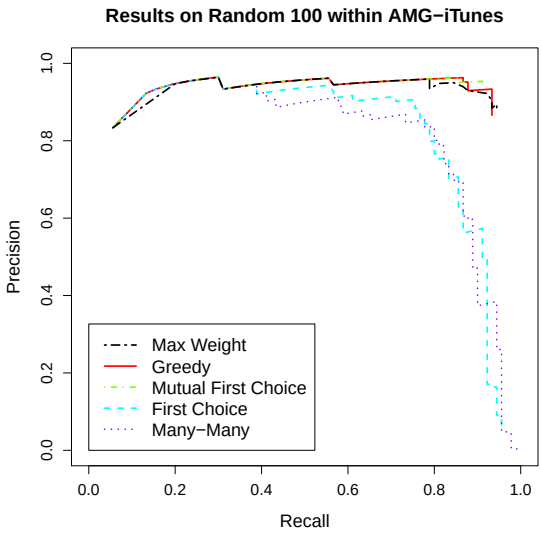

Figure 9: Results of testing the entity matching algorithms using IMDB-Netflix trained scores, on a truth set of random AMG-iTunes movies.

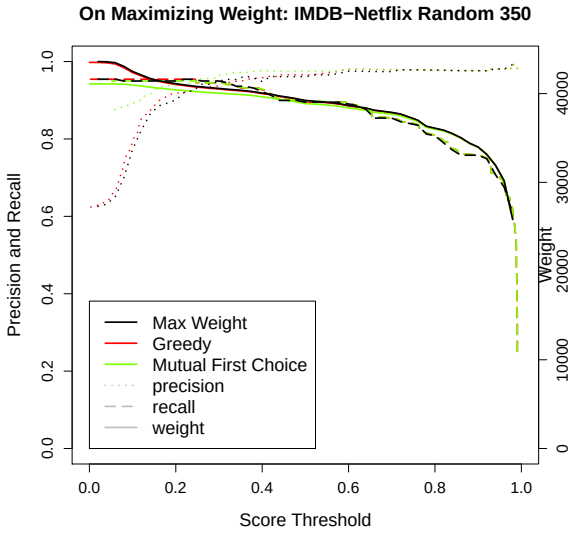

Figure 10: All three one-to-one algorithms perform as expected on the IMDb-Netflix random 350, with precision (recall/weight) increasing (decreasing) with threshold.

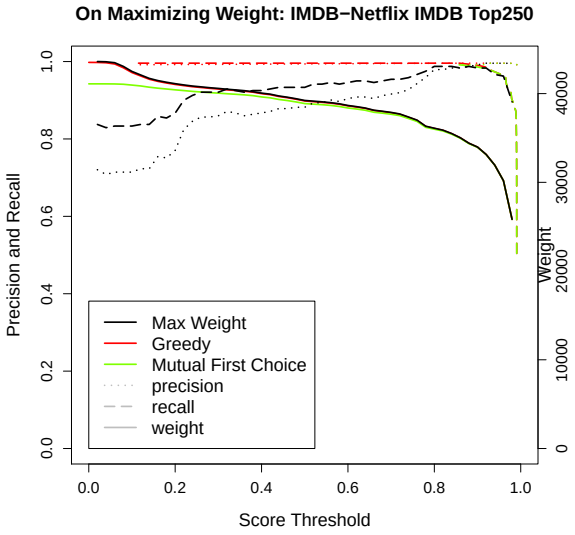

Figure 11: Max weight matching's recall is significantly lower than the other one-to-one algorithms for trivial thresholds, and increases with increasing threshold on the IMDb-Netflix IMDb top 250.

29,871 pairs unmatched by Freebase. We took a random sample of 100 pairs from that set and found only one incorrect pair, for a precision of 99%. *Compared to Freebase, our algorithm achieved superior precision and significantly improved recall.*

4.4 Effects of Limited Score Functions

We conjectured that one-to-one constrained resolution would be the most robust to a less accurate score function. To verify our hypothesis, we evaluated the resolution algorithms on new data sources, but without re-training—we use the same score function trained on the IMDB-Netflix Boundary training dataset.

Figures 8 and 9 display the results of matching movies between the AMG and the Apple iTunes Store online movie catalogs. The score function is trained on the IMDB-Netflix data to test the resilience of one-to-one entity resolution under poorly tuned scoring. Figure 8 shows the results for popular movies and Figure 9 shows the result for a random set selected from iTunes manually matched to AMG. While all algorithms' performances degrade from before, it is clear that ManyMany suffers the most degradation while

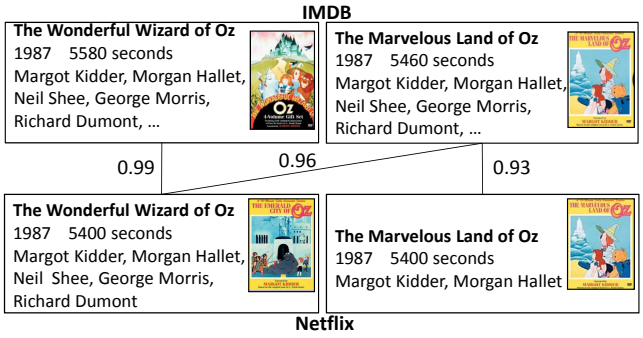

Figure 12: Example scores. FirstChoice adopts the left two arcs. MutualFirstChoice adopts only the leftmost arc. Greedy correctly matches both movies without the false positive.

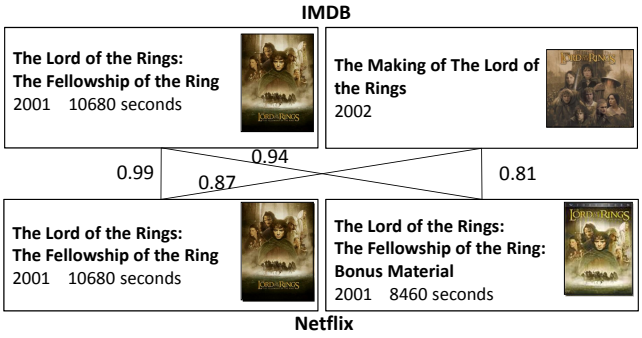

Figure 13: Example of scores that can mislead max-weight matching. In this case, max-weight adopts the diagonal arcs.

the three one-to-one algorithms are most robust. This is particularly true in the case of the set of random movie pairs, in which resolution is generally more difficult. This positive result has been repeated with other sources in subsequent work for the production system.

We have found that *the one-to-one constraint can make resolution more robust to the score function.* This implies that practitioners wanting to perform entity resolution can put less effort into the underlying scoring if they make use of one-to-one constraints, simplifying development effort and improving statistical performance.

5. DISCUSSION

5.1 Comparing Constraints by Example

We have seen that one-to-one constraints can improve aggregate precision/recall, but it is helpful to look in more detail at examples in order to understand the differences between the resolution algorithms. First we can look at how constraints solve problems that occur in the unconstrained case. Consider the movies *Die Hard* and *Die Hard 2*, released in 1988 and 1990 respectively, and with many of the same cast members. With such similar titles, release years and casts, they will have a very high score when compared together. The unconstrained ManyMany algorithm will thus produce four matches (each movie will match with itself and the other) where there are really only two.

The one-to-many constraint is illustrated in Figure 12, which shows the cartoon movies *The Wonderful Wizard of Oz* and *The Marvelous Land of Oz*, both released in 1987, with the same director, the same cast and a very similar runtime. Note that the diagonal arc has a higher score than the arc on the right because the greater number of matching cast members outweighs the lack of exact title match (the bottom right movie only lists two cast members). In this case, FirstChoice would match the Netflix *The Wonderful Wizard of Oz* with both IMDB movies, getting one match right but also one false negative and one false positive. MutualFirstChoice would only match *The Wonderful Wizard of Oz* between IMDB and Netflix, but not match *The Marvelous Land of Oz* for one false negative. Greedy would correctly match both movies and ignore the erroneous arc.

MaxWeight can be misled by a number of very similar movies. In Figure 13, all four movies have similar cast members, years and titles. The "Making of" and "Bonus Material" movies are closer in their titles to the full movie than each other, causing the sum of scores on the two diagonal arcs (incorrectly matching them to the full movie) to be greater than the sum of the vertical arcs (which includes the correct match of the full movie).

5.2 On the Role of Weight Maximization

Out of the six comparisons of the one-to-one algorithms' Precision-Recall curves, MaxWeight exhibited seemingly unreasonable behavior in two: its curve begins with high precision and recall for high score thresholds, and then doubles back on itself as the threshold decreases. Moreover this occurred only on the IMDB Top 250 and Netflix Top 100 evaluation sets under IMDB-Netflix resolution.

MaxWeight may produce drastically different results under small changes in the threshold, while the other one-to-many and one-to-one meta-algorithms' already resolved pairs must go unaffected. Consider Figure 13, which shows a pathological case on which MaxWeight fails. With a threshold between (0.94,0.99) the 0.99 edge is selected, corresponding to a true positive. When the threshold drops to below 0.87, however, the matching flips to produce two false positives, identifying feature films with one of two distinct bonus material entities. Since the bonus materials share little in common between their titles, their score together is low. Even though they could conceivably be resolved leaving the feature movies to be correctly resolved, the total weight of 1.80 is just shy of the 1.81 weight of the double false positives. Indeed this very situation occurs multiple times for the top movies on IMDB and Netflix, since such movies have associated documentaries and bonus materials cataloged by these two data sources. AMG/iTunes top movies, however, do not contain these bonus materials, perhaps due to a smaller catalog, and so MaxWeight achieves a maximum matching by selecting true positives. Finally, the movies contained in the random and boundary sets are not feature films so have no associated satellite movie entities.

Figures 10 and 11 further explore the interplay between

matching weight, precision and recall, for the three one-to-one meta-algorithms. The first figure shows three algorithms that achieve similar precisions, recalls and weights, with these increasing and decreasing respectively with increasing threshold, as is intuitively reasonable. The second figure shows one of the top movie evaluations, with MaxWeight's inferior recall increasing initially due to multiple pathological cases (*i.e.*, increasing true positives). However we see that for both figures, MaxWeight is rarely superior: in the former, MaxWeight's precision converges to best only at threshold 0.5.

Together the pathological case from popular movies, and the results on random tail movies, demonstrate that *weight maximization for entity resolution can fail miserably. What is needed are graph matching algorithms that optimize precision/recall.*

6. CONCLUSIONS & OPEN PROBLEMS

The concept of leveraging socio-economic properties has been fundamental in the history of web search. We are entering an era in which the Web is becoming increasingly structured, and new socio-economic properties will continue to play an important role. This paper first introduces the property that websites often have an incentive to avoid needless duplication of entities, and then leverages this property by applying one-to-one global constraints to significantly improve ER accuracy and robustness. We evaluate our generic ER framework on a large-scale movie matching task implemented in the movies vertical of a major search engine.

We show that on both head and tail movies, one-to-one resolution is superior to the unconstrained approach. Our many-to-many unconstrained algorithm is representative of other existing approaches: it involves blocking movie pairs to reduce the set of pairs that must be scored, movie attributes are individually scored, and a trained logistic regression model is used to combine these scores. Even with relatively unsophisticated feature-level scores, adding global constraints to the resolution achieves strong performance. We also show that constrained resolution is more resilient to a poorly trained score combiner, comparing the competing approaches by training on one pair of datasets before resolving a second pair of datasets. Finally we compare our one-to-one resolution to a crowd-sourced matching from Freebase, with the result that our algorithm enjoys improved precision and vastly improved recall.

A number of interesting open problems remain for future work. We showed that maximizing matching weight can be a poor surrogate for optimizing precision/recall. What is the link between weight and precision/recall? In particular, can the weights be calibrated so that weight maximization better aligns with our true objectives? Secondly, we have explored generic approaches to constrained resolution that work on top of existing many-to-many algorithms; how can pairs be scored and resolved simultaneously, as opposed to using such a two step process?

7. ACKNOWLEDGMENTS

We would like to thank Olivier Dabrowski, Craig Hunt, Piali Choudhury and Tristan Lahey for many helpful discussions and assistance relating to this research, Andrew Goldberg for his code used in our implementation of exact max weight graph matching, and the members of MSR Silicon Valley for their help in labeling data.

8. REFERENCES

[1] A. Arasu, R. Christopher, and D. Suciu. Large-scale deduplication with constraints using dedupalog. In *International Conference on Data Engineering*, pages 952–963, 2009.

[2] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom. Swoosh: A generic approach to entity resolution. *The VLDB Journal*, 18(1):255–276, Jan 2009.

[3] M. Bilenko and R. J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proc. KDD'03*, pages 39–48, 2003.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] S. Chaudhuri, A. Das Sarma, V. Ganti, and R. Kaushik. Leveraging aggregate constraints for deduplication. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 437–448, New York, NY, USA, 2007. ACM.

[6] Z. Chen, D. V. Kalashnikov, and S. Mehrotra. Exploiting context analysis for combining multiple entity resolution systems. In *Proc. SIGMOD'09*, pages 207–218, 2009.

[7] P. Christen. Automatic training example selection for scalable unsupervised record linkage. In *Proc. PAKDD'08*, pages 511–518, 2008.

[8] W. W. Cohen, P. D. Ravikumar, and S. E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proc. IJCAI-03 Workshop IIWeb-03*, pages 73–78, 2003.

[9] A. Culotta and A. McCallum. Joint deduplication of multiple record types in relational data. In *Proc. CIKM'05*, pages 257–258, 2005.

[10] T. I. M. Database. IMDb top 250, 2010. `http://www.imdb.com/chart/top`.

[11] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. Duplicate record detection: A survey. *IEEE Trans. Knowledge and Data Eng.*, 19(1):1–16, 2007.

[12] I. P. Fellegi and A. B. Sunter. A theory of record linkage. *J. American Statistical Assoc.*, 64(328):1183–1210, 1969.

[13] M. A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida. *J. American Statistical Assoc.*, 84(406):414–420, 1989.

[14] J. M. Kleinberg. Hubs, authorities, and communities. *ACM Computing Surveys*, 31, 1999.

[15] H. Köpcke and E. Rahm. Training selection for tuning entity matching. In *Proc. Int. Workshop on Quality in Databases and Management of Uncertain Data*, pages 3–12, 2008.

[16] H. Köpcke and E. Rahm. Frameworks for entity matching: A comparison. *Data & Knowledge Eng.*, 69(2):197–210, 2010.

[17] T. M. Mitchell. Generative and discriminative classifiers: Naive bayes and logistic regression, 2010. Manuscript available at `http://www.cs.cm.edu/~tom/NewChapters.html`.

[18] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

[19] J. C. Pinheiro and D. X. Sun. Methods for linking and mining massive heterogeneous databases. In *Proc. KDD'98*, pages 309–313, 1998.

[20] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. ISBN 3-900051-07-0.

[21] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proc. KDD'02*, pages 269–278, 2002.

[22] A. Segev and A. Chatterjee. A framework for object matching in federated databases and its implementation. *Int. J. Cooperative Info. Sys.*, 5(1):73–99, 1996.

[23] W. Shen, X. Li, and A. Doan. Constraint-based entity matching. In *National Conference on Artificial Intelligence*, pages 862–867, 2005.

[24] W. Su, J. Wang, and F. H. Lochovsky. Record matching over query results from multiple web databases. *IEEE Transactions on Knowledge and Data Engineering*, 22:578–589, 2010.

[25] S. Tejada, C. A. Knoblock, and S. Minton. Learning domain-independent string transformation weights for high accuracy object identification. In *Proc. KDD'02*, pages 350–359, 2002.

[26] W. E. Winkler. Advanced methods for record linkage. In *Proc. Section on Survey Research Methods, American Statistical Assoc.*, pages 467–472, 1994.

[27] W. E. Winkler. Overview of record linkage and current research directions. Technical Report Statistics #2006-2, Statistical Research Division, U.S. Census Bureau, Washington, DC 20233, 2006.

[28] H. Zhao and S. Ram. Entity identification for heterogeneous database integration: a multiple classifier system approach and empirical evaluation. *Info. Sys.*, 30(2):119–132, 2005.

APPENDIX

A. GENERIC SCORING FOR ER

As described in Section 2, many-to-many entity resolution makes use of blocking to prune down the number of entity pairs to compare, feature-level scores for comparing entities' attributes, and score combination via machine learning [3, 6, 15, 16, 19, 21, 22, 25, 28]. This section describes this abstract process, and provides detail on the particular learning algorithm we use for combining feature-level scores.

Blocking. We adopt a two-step scoring system in which all pairs are first passed through a conservative blocking mechanism $g(x_1, x_2) \in \{0, 1\}$ that filters out pairs which are likely to score close to zero—the range corresponds to that of logistic regression which outputs scores $h(\cdot) \in (0, 1)$. We score these pairs zero without explicitly evaluating the learned score as is common in other approaches to entity resolution [27, 16]:

$$f(x_1, x_2) \quad = \quad g(x_1, x_2) \cdot h(x_1, x_2) \ .$$

By choosing a mechanism g that is cheap to evaluate, we stand to significantly cut down on the most expensive phase of resolution: scoring. We detail our domain-specific approach to blocking in Section 3.

Feature-Level Scores. We consider the domains of $\mathcal{D}_1$ and $\mathcal{D}_2$ to be the same. This presumes some normalization during data ingestion, and implies that one dataset may have all null values for a given attribute that it does not support. In these cases scoring $h(\cdot)$ involves first evaluating d feature-level scores $h_1(x_1^1, x_2^1), \ldots, h_d(x_1^d, x_2^d)$, and then combining them using a supervised machine learner [15, 28, 16]:

$$(x_1, x_2) \mapsto h\left(h_1(x_1^1, x_2^1), \ldots, h_d(x_1^d, x_2^d)\right) \ .$$

We discuss our specific feature-level scores in Section 3, however common choices involve string matching and L_p norms for numeric vectors [8, 16].

Logistic Regression. Logistic regression is a simple but effective method for supervised classification [4] used in several previous works on entity resolution to combine feature-level scores [22, 19, 28, 15, 6]. Let X be a random variable representing the feature-level scores of a pair of entities, and Y be a random Boolean-valued variable indicating whether the pair is a match. Logistic regression employs a parametric model of the posterior likelihood of the form

$$h(\mathbf{x}) \ = \ \Pr\left(Y = 1 \mid X = \mathbf{x}\right) \quad = \quad \frac{1}{1 + \exp(\mathbf{w} \cdot \mathbf{x} + w^0)} \ ,$$

where $\mathbf{w} \in \mathbb{R}^d$ and $w^0 \in \mathbb{R}$ are the model parameters. One interpretation of logistic regression, is as a result of a generative model under the Naïve Bayes assumption (class conditional independence of the features) in which the conditional likelihood functions are Gaussians. An important property of logistic regression, is that the classifier's decision boundary is linear, since a monotonic transformation of the log-odds is linear in the feature vector:

$$\log_e \frac{\Pr\left(Y = 1 \mid X = \mathbf{x}\right)}{\Pr\left(Y = 0 \mid X = \mathbf{x}\right)} \quad = \quad \mathbf{w} \cdot \mathbf{x} + w^0 \ .$$

To fit the model to a training set of Boolean-labeled feature vectors, the conditional data likelihood—the likelihood of the observed Y values conditioned on the observed features—is maximized via an iterative gradient ascent procedure. Parameter fitting in this way is more robust to incorrect modeling assumptions than Maximum Likelihood Estimation [17].

For each entity resolution algorithm, we first train a logistic regression model on a labeled set of entity pairs within and outside $\mathcal{R}$, of moderate size much smaller than $|\mathcal{D}_1| \times |\mathcal{D}_2|$. With this scoring function h in hand, we may proceed to score all or those unfiltered pairs of entities due to blocking, and continue with the other steps of the particular resolution algorithm. Note that the time complexity of evaluating h on a pair is efficient at $\Theta(d)$ if each feature-level score evaluation takes constant time; in reality the constant may be relatively large.