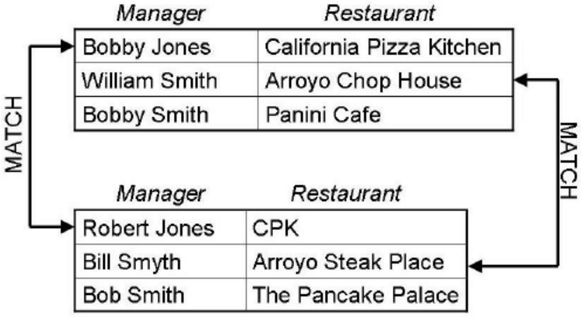
Record Linkage Using Heterogeneous Graph Techniques

Ittoop Shinu Shibu, Udaikaran Singh, Wesley Kwan

Problem Background

The problem we are looking to solve is the issue of record linkage, which is the process of relating entities within different datasets. An example of record linkage at a University would be relating the students within a department to the students within a class. This is made trivial by the fact that the school designates a distinct ID to each student. However, without this primary key, the task would become about using the features associated with each entry to relate the entities within each dataset.



Manager	*Restaurant*
Bobby Jones	California Pizza Kitchen
William Smith	Arroyo Chop House
Bobby Smith	Panini Cafe

Manager	*Restaurant*
Robert Jones	CPK
Bill Smyth	Arroyo Steak Place
Bob Smith	The Pancake Palace

Figure 1: Record Linkage Framework

The history of record linkage goes back to a paper written by Halbert Dunn in 1946 within the American Journal of Public Health. The original goal of record linkage was to be able to link the health records of patients from different sources. This paper laid out a high-level goal associated with the task of record linkage, and it was void of any model-based methods to link entities. Many years later, in 1959, Howard Newcombe laid out an automatic process of record linkage within the magazine [Science](https://science.sciencemag.org/content/130/3381/954. This article was the first to frame the problem of record linkage within a probabilistic framework, which is the perspective we will use in this project. This was later formalized by Ivan Fellegi and Alan Sunter.

There are two major categories of record linkage that are academically explored: (1) deterministic linkage and (2) probabilistic linkage. Both deterministic and probabilistic look to solve the problem of how we can link entities within two relations without the existence of a clearly discriminatory key, such as a social security number.

Deterministic linkage, which is more commonly used within the academic database community, is focused on the goal of creating a process to determine a primary key within a relation. In the cases that such a discriminating feature is absent, the process of making a composite primary key (and testing that this composite key is a valid primary key) is the goal of deterministic linkage. An example of this might be to look at a combination of (zip code, birth date, and last name) in lieu of SSN. Though this set of attributes are not inherently sufficient to discriminate between two people, it may be sufficient for a given relation.

On the other hand, probabilistic linkage looks to assign a probability to the event that 2 records belong to the same entity. This inherently allows for errors and works well with databases that are imperfectly linked as it looks to create probabilistic models that can be generalized to different sets of relations. As a result, the goal of probabilistic linkage focuses more on feature representation and making models rather than focusing on the columns of any specific dataset.

The primary uses for record linkage is entity resolution, which allows for efficient joining between multiple heterogeneous relations that do not have a clear discriminatory key and removes duplicates from a single relation. Although the usefulness of record linkage may seem limited in scope, the use of databases is pervasive within modern academic research and industry. Therefore, making progress in the problem of record linkage would likely impact all aspects of modern life through smaller incremental optimization that is hidden to most consumers.

Within the academic community, the main two fields of research that deal with the topic are the database community and the artificial intelligence community. Within the AI community, the focus is primarily on building probabilistic models, especially how approaches in machine learning, such as how Bayesian networks and multi-level perceptions can be used to improve on the power of such probabilistic models.

Proposed Solution

Traditional Framework The original framework presented by Fellegi & Sunter set up the problem of record linkage as assuming that there are two distinct sets (A and B), and we are trying to place pairs of records within the sets of M and U such that:

$$M = (a_i, b_j) : d_{ij} = 1, i = 1, \dots, n, and\ j = i, \dots, m$$
$$U = (a_i, b_j) : d_{ij} = 0, i = 1, \dots, n, and\ j = i, \dots, m$$

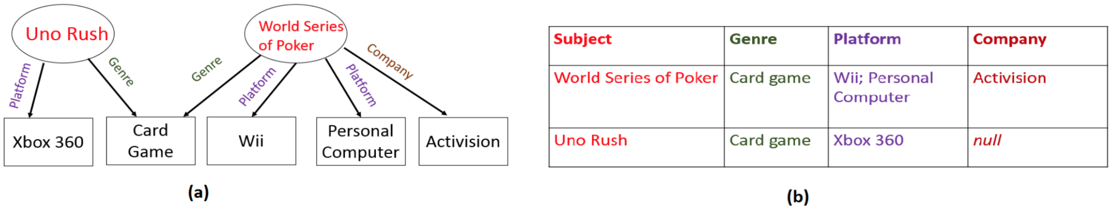
Then, based on the comparisons between the features of a_i and b_j, which is represented as g_{ij}, we would learn the probability distribution of $P(g_{ij}|D_{ij} = d_{ij})$.

The assumptions made within the model is that there are distinct (a_i, b_j) pairs within the dataset. Therefore, we are assuming that the characteristics of (a_i, b_j) must be mapped to either a 0 or 1 deterministically. However, given this assumption, the problem reduces to a binary classification problem. This framework lends itself to many of the traditional machine learning models that are commonly used, such as naive Bayes classifiers & SVM.

Our Framework Our framework will look to represent this machine learning problem as an edge prediction problem on a heterogeneous graph. Although the record linkage problem can be used between entities within the same relation, we will focus primarily on the task of joining two distinct relations. The reason for this is that it constrains the problem statement, allowing us to focus on the methodology before seeking for a generalizable solution. Our heterogeneous graph representation will essentially be layers of graphs. A fundamental graph within our representation is G_E, which is essentially all the nodes representing distinct enties within two relations. The edges for G_e are present if the two nodes represent the same entities. This graph is naturally a bipartite graph, because edges of nodes within the same relation cannot exist within our framework. Our prediction task will be predicting the edge on this graph while leveraging the encoded relationships between nodes encoded in our other graphs.

A second fundamental graph within our framework is G_{EA}. In this graph, there are 2 types of nodes: (1) entity nodes & (2) attribute value node. An edge in this case would be an ‘is a’ relationship that essentially encodes the tabular data. An example of this is in Figure 2.

Lastly, a third fundamental graph is G_{AA}. This graph has the node types of attributes and it connects different attributes. This graph, which can also be decomposed as different graphs for different relationships, is the most vague in our framework. This allows us to represent relationships between the



Subject	Genre	Platform	Company
World Series of Poker	Card game	Wii; Personal Computer	Activision
Uno Rush	Card game	Xbox 360	*null*

Figure 2: Example Graphical Representation

attribute values. For example, we could encode geographical data on a city nodes by linking them to a state node. Similarly, we could create links between categorical age nodes to a binning node that may be defined as ranges of ages. This allows for a natural representation of information that is beyond the data present in the dataset itself. A non-graphical interpretation of this is to create new columns within our tabular representation, which naturally does not scale well in terms of speed and space.

However, due to G_{AA} being very malleable, it can be created with a varying amount of precision and there is no apparent way to automate the creation of this graph.

The heterogeneous graphs will allow us to encode the data represented within the relation in a graphical manner. For example, if there is a categorical variable within each relation, then there can be an 'is a' edge that connects entities within each dataset. A quantitative variable will be treated differently, as ignoring it could result in generalization of data and including it could make it hard to create any edges between relations. Hence, we will be experimenting with various methods of manipulating quantitative variables such as binning, etc. to see how this impacts the edge prediction problem.

Therefore, we will be working to generalize all types of columns within the dataset to create edges between each relation. The general assumption that will follow within our model is that if the "strength" of a meta-path is high between the 2 nodes, then we can predict that there is an edge present, thus linking the records.

We believe that the graphical representation will improve on traditional models, because it allows for the expression of more complicated relationship between entities that a tabular model may not able to represent. Also, a graphical model is more modular, thus allowing for development within this project to be more easily generalizable. An example of how a graphical approach improves on a tabular is that adding multiple relations to the graph become more natural. Similarly, feature representation within a graphical framework is more flexible, such as representing an address based on its subparts (address, city, state, zip code). This allows for matching even in cases that there are errors within the relation, such as incorrect spelling or missing data.

Also, the graphical approach allows us to leverage previous research within the field, such as using the techniques described in node2vec. An example could be using the idea of negative sampling to improve the size of the training data.

Relationship to Domain

Record linkage is a problem that has a large amount of weight today. There are many ways to solve this problem, but when it comes to dealing with large datasets, working with graphs is the best option as it preserves a lot of the complex relationships that are lost when using other techniques. Depending on the datasets we choose to work on, we will be breaking them down into different node types that are connected by certain relationships.

An example of this would be converting three datasets taken from a medical institute that stores general patient information such as weight, height, and name in one dataset, patients diagnosis, conditions and name in another, and patient drug information and name in the last dataset. These datasets would be converted into a heterogenous graph with three different node types, diagnosis, drugs and anatomy. The key factor about these datasets is that they don't have a deterministic primary key such as ID to easily link the nodes. Instead, they are linked using a column that they might share, such as date of birth or name. This would result in a lot of duplicate entries and there could also be clerical mistakes (such as typos) that result in dangling references between nodes.

After converting our datasets into a heterogenous graph, we would implement various machine learning models to try to join the datasets with the highest accuracy to remove the duplicate entries and take care of the dangling references. This will largely depend on similarity measures and probability as the edge weights between the nodes will be based on the 'strength' of the relationship between the individual nodes from the heterogenous graph.

Relationship to Previous Work

Our work is heavily dependent on previous work done by Ivan Fellegi and Alan Sunter. We will be using the two methods talked about in the paper, deterministic linkage and probabilistic linkage, in order to create the links between nodes. The deterministic linkage tries to create a process to find a possible primary key within a given dataset. Since the datasets we would be dealing with lack a primary key, we would have to create a primary using a combination of the individual columns and how they are related to each other. The probabilistic linkage deals with assigning a probability of two records being part of the same unit. We will be using both these techniques in order to create our heterogenous graphs.

Our data cleaning and methods of deriving the nodes will stem from the following paper, Building a National HIV Cohort from Routine Laboratory Data: Probabilistic Record-Linkage with Graphs It details how we can create the nodes for our graphs by understanding the relationships in our datasets and create our own training data for our machine learning model. We will use An efficient record linkage scheme using graphical analysis for identifier error detection in order to understand how to deal with dangling references and duplicate records.

Our project will combine the ideas from the different papers mentioned above in order to create try to create a better solution to the record linkage problem. We also look forward to testing multiple models in order to see if there is a better classifier that can solve the problem.

Since we worked with heterogeneous information networks in malware detection, using heterogeneous information networks for record linkage is a clear extension of our previous work. We are introducing something new by tackling a problem unrelated to the cybersecurity space.

Our Approach

We will be using two datasets for our project. The first dataset we will be using is a donated dataset from Epidemiologisches Krebsregister NRW, which is a German cancer registry. This data set contains records with attributes such as name, sex, date of birth, and postal code. Data pairs in this data set are classified as match or non-match from a manual review that was conducted when this data set was made. Of the 5,749,132 record pairs, 20,931 are matches. Hence, it has a lot of noise and weak key pairs. Since this data is readily available of UC Irvine's Machine Learning Repository, data ingestion will be minimal. Our focus will be on cleaning the data and performing splits on the dataset for training our models.

The next dataset will be a fake generated dataset. We will be constructing this dataset by creating or taking a dataset with strong keys and breaking down the strong keys into different attributes with added noise. Hence, this dataset should help us understand if our model is able to make accurate edge

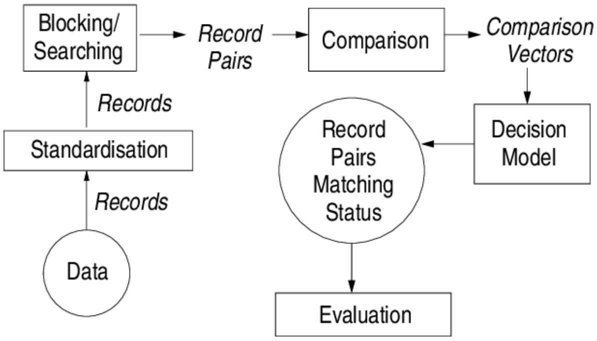


Figure 3: High-Level Steps

predictions by comparing the strong key dataset with the result of our model on the fake-generated dataset.

We will use feature representations with heterogeneous information networks, as we did with malware. As explained above, we will be creating graphs with differing node types and connecting them based on relationships we define. The labels on the data were primarily generated by matching phonetic equality of names; since we cannot extend on this, we can instead find an equivalent by analysing the text in names. This can be further improved on by connecting relationships between name and the other attributes of the data set. After defining our graphs, we will feed them into a classifier - for this, we can use Naive Bayes, logistic regression, SVM, etc. Our baseline model will be a Naive Bayes Model. To evaluate our model, we will use a variety of metrics. Looking at the record pairs, we can see that this is an imbalanced binary classification task - because of this, we will be using metrics such as BCR and F-score to better gauge our model's performance.

Project Output

Since this problem is found in the data science academia community, our project output will likely be a paper. We will produce the paper in a similar fashion to the paper replication project we did this quarter. First, there will be a section dedicated to discussing the data set used, the exploratory data analysis, and decisions used to clean the data. Then, we will discuss our methods and how we used heterogeneous information networks in the context of the record linkage problem. This will include explanations on the features and graphs we defined and used to train our classifier. Lastly, the paper will discuss results and whether these information networks made non-trivial progression on the record linkage problem. Since a paper is not always the most readable or easily understood method of communicating results, we may condense our process and findings into a project poster, which can visually display condensed snippets of our paper.

Works Cited:

1. https://ajph.aphapublications.org/toc/ajph/36/12
2. https://science.sciencemag.org/content/130/3381/954
3. https://www.tandfonline.com/doi/abs/10.1080/01621459.1969.10501049
4. https://aspe.hhs.gov/report/studies-welfare-populations-data-collection-and-research-issues/two-methods-linking-probabilistic-and-deterministic-record-linkage-methods
5. http://ceur-ws.org/Vol-1272/paper_17.pdf
6. https://archive.ics.uci.edu/ml/datasets/Record+Linkage+Comparison+Patterns?fbclid=IwAR1dLbJZGVJ4at79aw4h
   zWOduLXJA8G9tqbgpWwmYimnk-mPMAnUg
7. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3039555/
8. https://pdfs.semanticscholar.org/2404/eb5760ec2925c075c7968c845d2cc6fda73b.pdf
9. http://sites.bu.edu/jbor/files/2018/10/Building-the-Cohort-10oct2018-1.pdf

Schedule

1. Week 1:
   - Revised Proposal
2. Week 2:
   - Shinu: Clean dataset using the EDA from previous week
   - Udai: Create code for creating generated datasets with Strong Keys
     - allow for a hyperparameter of varying noise.
   - Wesley: Finalize Structure of our Heterogeneous Graph
     - Determine how we are going to represent quantitative variables
3. Week 3:
   - Create Heterogeneous Graphs on our datasets
   - Research machine learning methods on graphs
     - primarily focus on feature representation
4. Week 4:
   - Test out different feature representations
   - Test out different ML models
     - tune hyperparameters
   - Goal: have a working pipeline from the table -> results
5. Week 5:
   - Evaluate our pipeline over different datasets
     - example: changing noise in our generated dataset
     - evaluating results on 2 “real” datasets
6. Week 6:
   - Create Report
   - Create Website
   - Final testing of the pipeline

Backlog:

- For Checkpoint 1:
  - All members - revised propsal
  - Udai: Create code for generating datasets with strong keys (and varying noise)
  - Shinu: Clean and perform EDA on our real-world datasets
  - Wesley: Finalize Structure of our Heterogenous Graphs
    - determine method for representing graphs (on disk)
    - determine method for using quantitative attributes.