Entity Resolution Using Classifier and HNSW Index for Large Scale Company Data Integration

Jiayi Chen
MSc Applied Data Science, Utrecht University

30 June 2023

Supervisor: Ramon Rico
Second Examiner: Kees Bosch

Abstract

Entity resolution is a key task in data integration aimed at identifying and linking representations of the same entity in multiple datasets. Traditional entity resolution methods usually face significant challenges due to high computational cost and data heterogeneity. This study proposes a new approach that combines supervised learning classifiers and hierarchical navigable small world (HNSW) indexing to address these challenges. Specifically, we convert the entity resolution problem into a binary classification task by utilizing XGBoost and Random Forest classifiers to predict record pair matching based on similarity features. In addition, we use BERT embedding to generate high-dimensional vector representations of text fields and use HNSW indexing for efficient nearest neighbor search. Our approach exhibits high accuracy and scalability, significantly reduces computational complexity and improves matching performance on large-scale datasets. The results highlight the potential of combining advanced machine learning techniques with efficient indexing structures to enhance entity resolution in different data environments.

Contents

1 Introduction

Entity resolution is a fundamental task in numerous applications such as natural language processing, aimed at identifying and linking representations of the same entity or object across multiple datasets or within a single dataset. A substantial body of research is dedicated to entity resolution; however, due to the variability in data patterns and the inconsistent quality of data, general models often fail to produce optimal results. Furthermore, traditional entity resolution methods are typically batch-processing tasks that analyze the entire dataset, which incurs significant computational costs. In classical pairwise comparison methods for entity resolution, it is challenging to avoid the $O(n^2)$ complexity of linking every record from one table with every record from another table. Also, given the prevalence of large-scale datasets nowadays, there is a critical need for effective solutions tailored to specific data contexts.

In the context of this study, we seek to identify entities within a dataset comprising nearly one million company profiles. In the Unknown Groups Venture Portal, comprehensive startup profiles are created by merging multiple startup databases. The integration of databases from various sources results in a heterogeneous and voluminous dataset, which complicates the resolution process. This challenge is exacerbated by the fact that attributes such as names alone are insufficient to confirm entity identity. Matching based on LinkedIn, website, and Twitter URLs remains unreliable and inconsistent, as some URLs may be inferred from the startup's name.

To address this issue, the startup currently employs the 'Splink probabilistic record linkage' method. This technique assigns a matching score to each company attribute based on its frequency in the database, thereby providing greater confidence when matching records with rarer attributes. Although this approach is effective at present, there remains significant potential for further improvement. For example, Splink is more suitable for handling datasets where the attributes are not highly correlated. However, it is conceivable that within the context of this dataset, the domain portions of different social media and website URL fields may exhibit a high degree of overlap. This is possibly because companies often name their various social media accounts with identical or similar names.

In order to solve the problem of incomplete data (many missing values) in URL text fields and large-scale data search, this study proposes an efficient entity resolution model based on a supervised learning classifier and combining transformer with HNSW indexing, and the following points are divided into detailed description of the ideas of this study dedicated to solving the problem:

1. Transforming the entity resolution problem into a problem of binary classification machine learning model as it is essentially determining whether pairs of records point to the same underlying entity. Learning decision boundaries based on features extracted from the records with the aim of categorizing record pairs as matching or mismatching. Data integration, deduplication and information retrieval tasks in various domains are facilitated by utilizing machine learning algorithms to predict the similarity or sameness between data set entities.

2. Systematically learning from examples of record pairs labeled as matching or mismatching entities through supervised learning, an approach that allows discriminative features to be extracted from the data, enabling the model to generalize and make accurate predictions about unseen record pairs.

3. Constructing training datasets for entity resolution using rule-based links. A systematic approach is provided to build basic factual labels for training datasets in entity resolution tasks by basing them on shared attributes. This approach not only facilitates the training of supervised learning algorithms by providing labeled examples, but also allows entity resolution techniques to be evaluated and improved through systematic validation against the ground truth.

4. The combination of generating text field embeddings using BERT and utilizing HNSW indexing optimizes embedding representation and search structure, thereby circumventing the costs associated with repeated complex text processing and similarity computations in entity resolution. This approach effectively streamlines the computational complexities involved in entity resolution tasks.

2 Related Work and Key Concepts Overview

2.1 Supervised Learning

Fully supervised algorithms require either a large amount of carefully selected training data or a large amount of reliable training data due to the high degree of class imbalance in evaluating record pairings (where matching pairs are quite sparse).

Training data for these algorithms are mostly collected through crowdsourcing (e.g., J.Wang, 2012) or laborious manual record linking exercises (M. J. Bailey, 2020). However, it is not always possible to collect credible training data needed to train complex learning algorithms such as deep neural networks in entity resolution tasks.

For example, Kooli et al. (N. Kooli, 2018) trained deep neural networks using approximately 10 million labeled record pairs, which equates to over 3,000 resolved individual recordings. Active learning and transfer learning have been investigated recently by Kasai et al. (J. Kasai, 2020) to train deep neural networks for entity resolution with fewer labels. When it comes to entity resolution for unstructured or textual problems, deep learning techniques are especially promising because pretrained language models can be applied (Y. Li, 2020). Simpler classifiers like logistic regression, decision trees, random forests, and Bayesian additive regression trees are typically favored for structured entity resolution (T. Hastie, 2001).

An illustration of the practical use of these techniques is provided by Ventura et al.'s work (P. Azoulay,2012), which focuses on a case study of inventor disambiguation in the patent bibliographic database of the United States Patent and Trademark Office (USPTO). The authors suggested a random forest-based supervised deduplication technique. Their training data came from prior research on "superstar" scholars in the living sciences and from the resumes of optometry innovators. This enabled them to train their random forest classifier on labeled comparison vectors of record pairings and assess the effectiveness of previously employed approaches in this application (G. C. Li, 2014)and to use labeled comparison vectors of record pairings to train their random forest classifier. They then used a four-step procedure to apply their entity resolution method to additional records in the USPTO database. They started by computing comparison vectors for each record pair inside each block using blocking techniques. Second, the matching probability for these comparison vectors was calculated by the authors using a random forest classifier. Third, discrepancy estimates between each pair of records were created using the projected probabilities. Fourth, utilizing single-link hierarchical clustering in accordance with the discrepancy scores from the preceding stage, the authors enforced transitive closure between record pairings.

Cutting the dendrogram (tree) at a threshold allowed for the determination of clusters. The final collection of duplicate records was obtained by combining all cluster results across blocks. The "Entity Resolution as a Clustering Problem" section will cover this kind of clustering technique, which can be applied either directly or as a step after pairwise classification.

2.2 Probablistics Record Linkage

Probabilistic record linking attempts to take potentially multiple fields, compute a weight for each field based on the estimated ability to correctly identify a match or a mismatch, and compute the probability of recording whether two records point to the same entity based on the weight. A threshold is set, and if that probability is higher than this threshold then the record is considered to be a match, and if the probability is lower than another threshold the record is considered to be a mismatch; pairs of records between these two thresholds have a probability of being a match. The complexity of probabilistic record linking is due to errors in the key of the link and the lack of a unique key to connect two records together. Most of the record linking research is based on the Fellegi-Sunter statistical framework, which centers on: similarity and thresholds. The model works as follows:

1. vectorized features: each attribute of each record is converted into a feature, and multiple features form a feature vector.

2. Similarity calculation: by comparing the feature vectors of two records, the similarity score between them is calculated.

3. Threshold Setting: Set a threshold for the similarity of whether the records are similar or not.

4. decision rule: based on the relationship between the similarity score and the threshold value, a decision is made whether to match the two records as consenting entities or not.

Movaffaghi et al. proposed an improvement on this model due to the fact that the Fellegi-Sunter model traditionally does not take into account the frequency of specific values in the matching field.

For example, common and rare values are treated equally, which can lead to inefficiencies. A method is proposed to adjust the matching weights based on the frequency distribution of values in the matching field. This adjustment helps to differentiate between common and rare values, resulting in more accurate matches. For example, two records sharing a rare surname are more likely to be a true match than records sharing a common surname. This involves calculating the probability of agreement for each possible value of the string variable and adjusting the weights accordingly. The method utilizes log odds to quantify match weight adjustments and ensure that frequency information is appropriately incorporated into the matching process. As well, it introduces the use of a log-linear model that incorporates the interactions between the variables. By doing so, it captures the dependencies between the matching variables, resulting in more accurate matching results.

2.3 XGBoost Classifier

XGBoost is proposed by Chen and Guestrin in 2016 and quickly gained popularity due to its effectiveness in various machine learning competitions.

Boosting is an ensemble machine learning technique that combines the predictions of multiple base estimators to improve the robustness of a single estimator. The main idea is to train predictors sequentially, with each predictor trying to correct the errors of its predecessor. Gradient boosting, the algorithm on which XGBoost is based, is particularly effective for both regression and classification tasks.

XGBoost includes several advanced features to enhance its performance and reliability. One key feature is its inclusion of L1 and L2 regularization, which helps to avoid overfitting, a common problem in many machine learning models. Additionally, XGBoost is designed to utilize parallel processing, enabling it to handle large datasets more efficiently by distributing the computational workload. Another important aspect of XGBoost is its tree pruning technique, specifically depth-wise tree pruning, which makes the learned trees more generalized and robust. Furthermore, XGBoost has built-in mechanisms to handle missing values during training, ensuring that the model remains effective even when some data points are incomplete. Lastly, each iteration of XGBoost includes built-in cross-validation to assess performance and mitigate overfitting, providing a reliable measure of the model's accuracy and stability.

2.4 Random Forest Classifier

Random Forests (RF) is an ensemble learning method used for classification and regression tasks. It builds multiple decision trees during the training phase and outputs the mode of the classes (classification) or the mean prediction (regression) of the individual trees during the testing phase to make the final prediction. Based on the concept of ensemble learning, Random Forest combines multiple models (i.e., decision trees) to solve specific problems, thereby enhancing the model's performance. Each tree is generated by bootstrap sampling (i.e., sampling with replacement) from the original data, meaning some data points may be repeated in the training set while others may be omitted (referred to as out-of-bag samples). At each node, a subset of features is randomly selected, and the best split is chosen among these features. This random feature selection helps to decorrelate the trees, reducing the variance of the ensemble. For classification tasks, the final prediction is based on the majority vote from all individual trees, while for regression tasks, the prediction is the average of all individual trees' predictions.

Random Forest has many advantages, such as being less prone to overfitting, robust to noisy data, capable of handling high-dimensional datasets, and providing estimates of feature importance. The steps to build a Random Forest include creating multiple bootstrap samples, generating a decision tree from each sample, randomly selecting a subset of features at each node, and choosing the best split based on these features, repeating this process until the trees can no longer grow. Finally, the predictions of all trees are aggregated by voting or averaging to make the final decision. Random Forest excels in handling high-dimensional data, providing high accuracy, and offering flexibility, making it suitable for both classification and regression tasks.

It combines the simplicity and flexibility of decision trees into a powerful and versatile method that is less prone to overfitting compared to a single decision tree and provides a good balance between bias.

2.5 BERT transformer

BERT (Bidirectional Encoder Representations from Transformers) is a language representation model proposed by Jacob Devlin and his team at Google AI Language. Unlike traditional language models, BERT pre-trains deep bidirectional representations by considering left and right context simultaneously in all layers.

The pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for various tasks, such as question answering and language inference, without substantial task-specific architecture modifications. BERT uses masked language models (MLM) and next sentence prediction (NSP) tasks for pre-training, enabling the learning of deep bidirectional representations. This model achieves new state-of-the-art results on eleven NLP tasks, including the GLUE benchmark, MultiNLI, and SQuAD, significantly reducing the need for task-specific architectures.

The architecture of BERT is based on the Transformer, including BERTBASE (12 layers, 768 hidden units, 12 attention heads) and BERTLARGE (24 layers, 1024 hidden units, 16 attention heads), and uses WordPiece embeddings with a vocabulary of 30,000 tokens. BERT's success demonstrates the importance of bidirectional pre-training for language representations, significantly enhancing the performance of NLP.

2.6 HNSW Algorithm

Hierarchical Navigable Small World (HNSW) is an algorithm designed for efficient and robust approximate nearest neighbor search, building on the concept of navigable small world graphs, which enable logarithmic scaling in the number of hops during graph traversal, making the search process highly efficient. HNSW creates a multi-layer graph where each layer is a proximity graph containing subsets of the data points, with the highest layer containing the sparsest graph with long-range connections, and each subsequent lower layer containing denser graphs with shorter-range connections. The elements are distributed across multiple layers, with the maximum layer an element can appear in being determined randomly with an exponentially decaying probability.

This hierarchical arrangement helps in efficiently narrowing down the search space. The search begins at the topmost layer, leveraging the long-range connections to quickly approximate the nearest neighbor. Once a local minimum is reached at a higher layer, the search continues at the next lower layer using the local neighbors found in the previous layer, and so on, until it reaches the lowest layer.

HNSW uses a greedy algorithm for graph traversal, selecting the closest neighbor to the query point at each step, and employs a heuristic to enhance performance, especially in highly clustered data, by considering the distances between candidate elements and the already connected elements to maintain diverse connections.

HNSW achieves logarithmic complexity scaling with respect to the number of elements, making it suitable for large-scale datasets. The use of hierarchical layers and greedy search ensures efficient processing even for high-dimensional data. During the construction phase, new elements are inserted into the graph in a way that maintains the hierarchical structure, involving selecting a random maximum layer for the element, performing a greedy search at each layer to find its neighbors, and establishing bidirectional connections. The memory consumption of HNSW is controlled by the number of connections per element in each layer, being memory-efficient while maintaining high search performance, showing significant improvements over other state-of-the-art methods in terms of query time and accuracy.

HNSW is highly effective for tasks requiring fast and accurate approximate nearest neighbor searches, such as image feature matching, semantic document retrieval, and various non-parametric machine learning applications, outperforming previous methods in both speed and accuracy across different types of datasets.

3 Data Description

The source of the dataset is mainly a scalable dataset about existing company information from a certain organization, with nearly one million data records.The original data is stored in BigQuery.BigQuery is a fully managed, serverless data warehouse provided by Google Cloud Platform (GCP) It primarily includes social media information for companies, such as Facebook, Twitter, LinkedIn, Instagram URL links, Crunchbase URLs, companies' homepage URLs, along with phone numbers and unique IDs for each record. However, each company may contain multiple records. Each data entry comes from different sources. Since the original dataset contains URL fields, the standardized dataset has processed these URL fields by removing protocols and symbols, retaining only the domain, subdomain, and query parameters.The attributes remain as follow:

- **id**: The unique identifier for a company or record. A hashed string used to uniquely identify each record in the dataset.
- **phone**: The primary contact telephone number for the company.
- **crunchbase_url**: Crunchbase is a company that provides information of businesses. The url of each company contains its investment and funding information.
- **homepage_url**: The company's official website, providing details about products, services, and information about the company.
- **linkedin_url**: A professional and employment-focused social media, for almost every company, it contains industry news and its activities.
- **twitter_url**: A social media platform used for sharing real-time information, brand reputation, and interactions.
- **facebook_url**: A social media platform used for brand promotion, product advertising, and community interaction.

The word clouds below provides a more intuitive understanding of the most common words or phrases within each field.

(a) crunchbase_url (b) facebook_url (c) homepage_url

Figure 1: Word Cloud of Some Fields

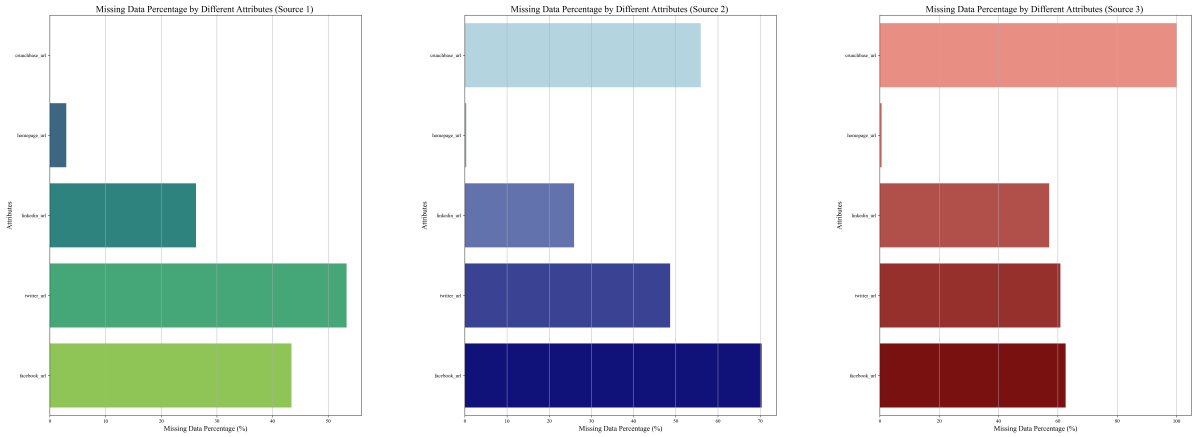
In addition, the languages included in the dataset are listed in the appendix.

There are a total of 545,255 records from source 1, and 424,743 records from source 2 and source 3 combined respectively. To facilitate subsequent data comparison, the data from source 2 and source 3 have been merged into a single dataset. Since each record's ID is unique, this merging operation will not cause any data duplication or loss, even though in Entity Resolution (ER), some records may refer to the same entity.

Handling Missing Values: The presence of missing values can notably affect the performance of the entity resolution model. The missing value status for each field is depicted in the following figure. However, it is imperative not to discard entire records simply due to the absence of certain fields, as this would compromise the integrity of our final results.

The correlation tells similarity between different fields, ranging from 0.12 to 0.18 indicate the Jaro-Winkler similarity score between different URL fields. These values are relatively low, indicating that the URLs in these fields are not very relevant.

The code ensures the confidentiality of sensitive data through robust encryption and access management protocols.We acknowledge responsibility for any potential misuse or unintended outcomes resulting from the code or project.



(a) (b) (c)

Figure 2: Missing Data Percentage by Field

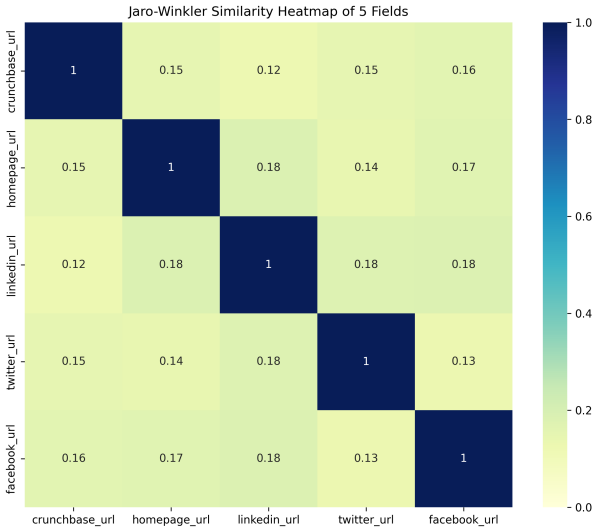


Figure 3: Correlation Matrix Heatmap of Jaro-Winkler Similarity Between Fields

4 Methodology

Entity resolution is a technique used to identify whether multiple data records in a dataset refer to the same real-world entity. In our scenario, we attempt to identify entities within a nearly 1 million company profiles dataset. However, databases from different sources result in a heterogeneous large amount of data that adds to the complexity.To translate this practical problem into a data science problem, we focus on the following aspects:

Data Heterogeneity: Different databases contribute to a wide variety of attribute formats and languages. Examples include spacers between domain names, dots between domain names and suffixes, leading to inconsistent and missing information. The data science challenge here is to develop methods to preprocess and standardize these heterogeneous data sources effectively.

Scalability: With nearly one million records, the data science question involves finding scalable algorithms and techniques that can handle large-scale data without compromising on accuracy and performance.

Pairwise Comparison Complexity: The traditional approach to entity resolution involves pairwise comparisons, which is computationally expensive for large datasets. Thus, the challenge is to devise efficient methods that reduce the computational complexity of these comparisons.

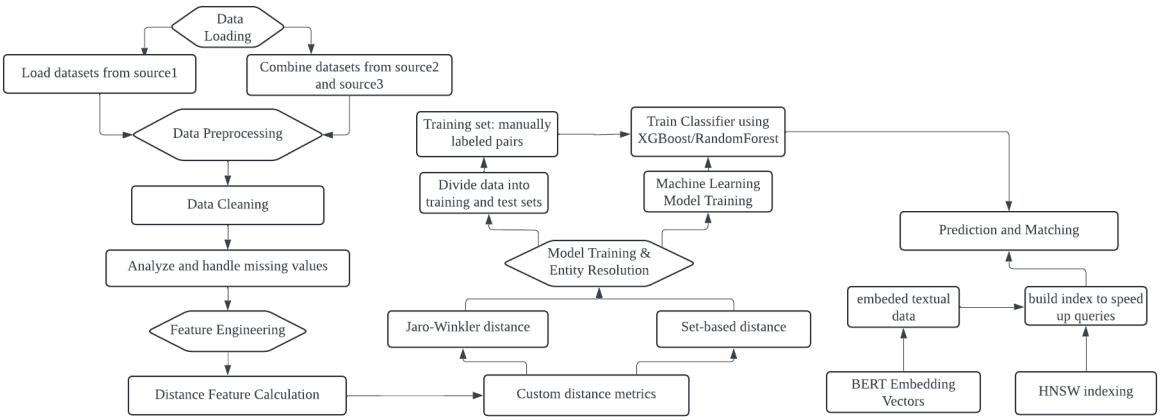


Figure 4: Technical Flow Chart

By addressing these aspects, we transform the practical problem of entity resolution in a large, heterogeneous dataset into a structured data science problem.Entity resolution can be seen as a binary classification task. Therefore, XGBoost classifier is chosen to train a model by learning similarity distance features of various attributes. Handling a large number of predictions to determine if entities are the same actually poses an $O(n^2)$ challenge.

Hence, we employ BERT to obtain word embedding vectors for paired records and HNSW (Hierarchical Navigable Small World) for rapid nearest neighbor search in an n-dimensional space, establishing a new index based on embeddings.

Training data preparation: Clearly, supervised learning requires a training dataset. To simplify the enormous workload of manually labeling data for training, we employ rules based on entity attributes to determine if two entities match.

The matching criteria are as follows: 1) Share the same crunchbase_url; 2) Share the same homepage_url or crunchbase_url; 3) Share the same linkedin_url or crunchbase_url or homepage_url.Labeling process: By examining whether these attributes meet the matching criteria, we assign labels (1 indicates a match, 0 indicates no match). These labels are referred to as the "is_match" column, indicating whether two entities are considered to be the same company.This method is suitable for this scenario where the dataset lacks clear unique identifiers.

To ensure the robustness of the model, we test the predictions of the model using a training set consisting of different data.

In training set 1, there are 77,681 records with identical crunchbase_url and 75,000 nonmatching records.

In training set 2, there are 100,000 records where either crunchbase_url or homepage_url matches (each comprising 50

In training set 3, there are 100,000 records where either crunchbase_url, homepage_url, or linkedin_url matches (each comprising one-third), along with 75,000 non-matching records.

Feature engineering: After merging two tables by columns, Jaro-Winkler distance and set distance metrics are applied to several URL fields. Jaro-Winkler similarity is an algorithm used to assess the similarity between two strings, taking into account character order and character distance. Set distance calculates the similarity of two strings, v1 and v2, based on their set intersection relative to the minimum set length, after splitting the strings into sets using a specified delimiter. These distance metrics are utilized for training machine learning models, aiding in better understanding and predicting relationships and similarities between data instances. Matched instances are assigned a label of 1, while unmatched instances are labeled as 0.

Model Training: Due to the heterogeneous nature of the data sources in this entity resolution task, a significant amount of missing values is present. XGBoost's built-in missing value handling mechanism can automatically manage these missing values in company datasets without requiring complex preprocessing, ensuring the robustness of the model. Additionally, XGBoost's ability to perform incremental learning allows the model to be gradually updated, effectively handling large-scale data. First, the XGBoost model was trained on the similarity distance features and labels. During the training process, the model's parameters were tuned, and a maximum tree depth of 5 and 100 estimators were selected for the

final model. To avoid overfitting and enhance the model's generalization ability, ten-fold cross-validation was employed. Furthermore, a Random Forest model with the same parameters as the XGBoost model was trained to provide a comparative analysis. The results of both models, including metrics such as accuracy, were compared to evaluate their performance.

BERT Embedding Generation: To convert the textual fields of prediction data into high-dimensional vectors using BERT, the input text is first preprocessed. For instance, in the case of company data, various fields are concatenated into a single text string (here, we select the URL field as the predictive component to create indexing feature objects). This involves tokenization, token masking, and adding special tokens, transforming the text into a sequence of lexical units. Each lexical unit is mapped to a corresponding word vector. Consequently, the semantic vectors obtained for each field serve as the basis for indexing.

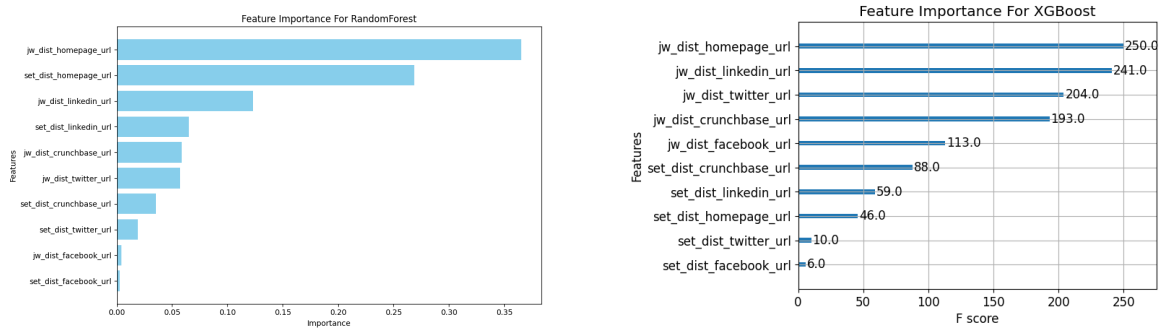
HNSW Indexing: The HNSW index p is initialized, specifying properties such as space type (Euclidean distance), vector dimensionality, and parameters 'ef_construction' and M during index construction. Embedding vectors for all company data are converted into arrays, and each company's index ID is obtained. Embedding vectors along with their corresponding index IDs are added to the HNSW index, thereby constructing an efficient structure for fast approximate nearest neighbor search in an n-dimensional space.

Prediction: On a real-world company dataset, we performed feature engineering on the URL fields, and based on nearest neighbor search and two distinct machine learning classifiers, predicted matches between two datasets. During this process, matches already known in the prediction set were filtered out.

5 Results

Model Performance: Through an extensive cross-validation procedure, we rigorously evaluated the performance and robustness of the XGBoost model using an independent test set with the 8:2 training-test split ratio.

Preliminary results from applying this testing framework show that the models developed using both XGBoost and Random Forest algorithms achieve an impressive 99% accuracy on the test set. This high accuracy rate highlights the ability of the models to correctly identify and match company record entities with high precision.



(a) (b)

Figure 5: Feature Importance Comparison

However, it is worth noting that there is a significant difference in the importance of individual features in the two algorithms. This difference highlights the different ways in which XGBoost and Random Forest utilize and prioritize features to achieve high performance, suggesting that each algorithm has a unique approach to processing and interpreting data.

In the XGBoost classifier model, the feature importance of the distance similarity between linkedin_url and homepage_url is the highest, which could mean that the degree of similarity between an entity's (e.g., a company's) LinkedIn and homepage URLs is an important differentiator in your dataset. For example, if two entities have very similar LinkedIn and homepage URLs, they may represent the same entity or closely related entities. Similarly, in the Random Forest classifier model, the degree of similarity of the two homepage_url is an important distinguishing criterion, if two companies have homepage_url they are likely to be the same company or very similar companies.

The following ROC plots show the performance of different classifiers (XGBoost and Random Forest) on different datasets (Training Set 1, 2 and 3).

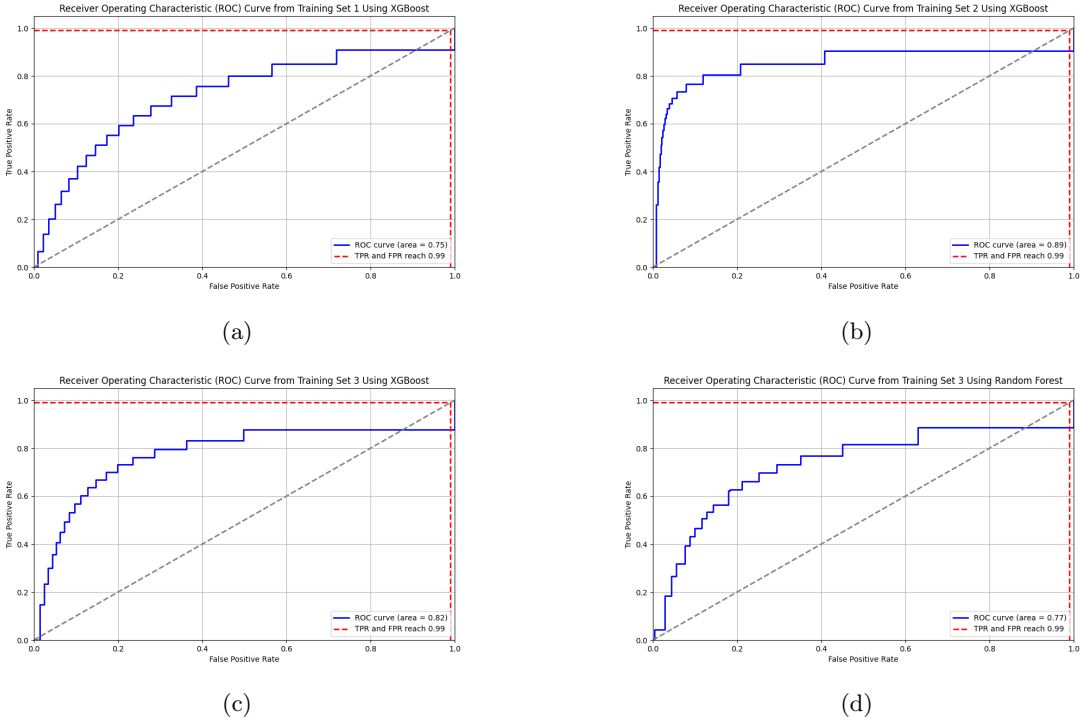


Figure 6: Combined ROC Curves

Performance on the different datasets showed variation, with the best performance on Training Set 2 (AUC = 0.89), and closer performance on Training Set 1 and Training Set 3 (AUC 0.75 and 0.82, respectively).

Random Forest performed better on Training Set 3 (AUC = 0.77), but not as well as the XGBoost model on the same dataset (AUC = 0.82).This suggests that for this dataset, the XGBoost model may be more suitable.

During entity match prediction, the system utilizes trained XGBoost and Random Forest (RF) models to predict potential matches, and it is worth noting that the performance of these models varies significantly across models. In addition, the use of Bert in conjunction with HNSW for nearest-neighbor search data matching greatly reduces the computational time overhead during the matching process.

Table 1: Comparison of Different Compositions of the Training Set and Matching Performance Across Two Models

Field Name/Model	**XGBoost (100%)**	**XGBoost (50%)**	**XGBoost (25%)**	**Random Forest (Default)**
crunchbase_url	100%	50%	25%	33.33%
homepage_url	-	50%	25%	33.33%
linkedin_url	-	-	50%	33.33%
Number of Matched Record Pairs	109,496	168,988	442,893	379,252

6 Conclusion and Discussion

The study systematically explored the application of supervised machine learning and transformer techniques to the task of scalable entity matching. The use of the HNSW (Hierarchical Navigable Small World) index significantly enhances query efficiency, greatly reducing average query times. From the HNSW theory's perspective, the HNSW index constructs a graph-based data structure that enables efficient approximate nearest neighbor search. This structure leverages the small-world phenomenon, where most nodes (representing data points) can be reached from any other node in a few steps. HNSW organizes these nodes hierarchically, creating multiple layers of graphs with increasing levels of connectivity.

In the top layers, the graph is sparse, facilitating quick traversal across large sections of the data space. As the search descends to lower layers, the graph becomes denser, allowing for precise local searches. This multi-layered approach reduces the search space exponentially as the query navigates through the layers, leading to substantial improvements in query times.

Furthermore, HNSW's dynamic nature allows for the continuous insertion and deletion of nodes without significant degradation of the index structure or search performance. This adaptability is crucial for scalable entity matching tasks, where the data is often large and frequently updated.

Although contextual BERT word embeddings are able to support multilingual texts, their capture of data features may not be sufficient under certain specific conditions. These conditions include the following three main features exhibited by the data space:

Large number of URL domain fields with low semantic information: the dataset contains a large number of URL domain fields that typically have low semantic information and may not provide sufficient contextual information for the word embedding model. This can limit the effectiveness of BERT in capturing and understanding the information conveyed by these fields, which in turn affects the performance of the model.

Most fields contain missing values: many fields in the dataset have missing values. This missingness causes the model to face the problem of insufficient data during training and prediction, which in turn affects the robustness and accuracy of the model. Although XGBoost and Random Forest can handle data with missing values, the handling of missing values by these models is not always guaranteed to be optimal, and the model performance may be significantly degraded in the presence of a high percentage of missing values.

Multiple languages detected in the data: although BERT is able to process multilingual text, its performance may be affected to some extent when confronted with datasets containing multiple languages. Grammatical structures and vocabulary usage vary greatly from language to language in a multilingual environment, which increases the difficulty for the model to accurately understand the semantics. In addition, some languages may receive less corpus during pre-training and thus show deficiencies in language-specific text processing.FastText can process context-independent text and is more suitable for this url field text, but no FastText pre-training model has been developed that can process multiple languages at the same time. In addition, supervised learning relies heavily on the characteristics and size of the training dataset. This means that the composition of the training set directly affects the predictive ability of the model, and the models as well as the results obtained from training vary from one training set to another, thus introducing certain limitations to the algorithm. If the data distribution in the training set does not adequately represent the data distribution in real application scenarios, the generalization ability of the model will be limited, which may lead to poor performance on real data.

To address these challenges, further research is necessary. These researches may include, but are not limited to, the following:

1. **Other form of comparison:** Entity Resolution uses a variety of methods for comparing multiple datasets in addition to pairwise comparison. For example , graph-based methods represent entities as nodes in a graph and resolve similar entities through community detection or clustering algorithms; integration-based methods synthesize information from different data sources for resolution. In addition, clustering algorithms such as k-means, Hierarchical Clustering (HCC), and Density-Based Clustering (DBSCAN) can also be used to group similar entities for comparison and parsing of multiple datasets.

2. **Improve model architecture:** Explore and develop new model architectures, such as using unsupervised learning methods such as clustering, because unsupervised learning can directly utilize a large amount of unlabeled data for training, which reduces the model's accuracy of model results due to the differences in the patterns of the training data.

3. **Single data source:** entity parsing of different companies can be increased by other fields of information about the company, e.g., address, name, etc. Increasing the source of the dataset can result in a training set with more complete information, which increases the accuracy of the record links.

By working on these aspects, we can expect to further advance the development of entity parsing techniques while improving the model's ability to handle complex datasets.

A Language Detection

Language Code	crunchbase_url	homepage_url	linkedin_url	twitter_url	facebook_url
af	✓	✓	✓	✓	✓
ca	✓	✓	✓	✓	✓
cs	✓	✓	✓	✓	✓
cy	✓	✓	✓	✓	✓
da	✓	✓	✓	✓	✓
de	✓	✓	✓	✓	✓
el	✓				
en	✓	✓	✓	✓	✓
es	✓	✓	✓	✓	✓
et	✓	✓	✓	✓	✓
fi	✓	✓	✓	✓	✓
fr	✓	✓	✓	✓	✓
hr	✓	✓	✓	✓	✓
hu	✓	✓	✓	✓	✓
id	✓	✓	✓	✓	✓
it	✓	✓	✓	✓	✓
ja			✓		
ko			✓		
lt	✓	✓	✓	✓	✓
lv	✓	✓	✓	✓	✓
mk			✓		
nl	✓	✓	✓	✓	✓
no	✓	✓	✓	✓	✓
pl	✓	✓	✓	✓	✓
pt	✓	✓	✓	✓	✓
ro	✓	✓	✓	✓	✓
ru			✓		✓
sk	✓	✓	✓	✓	✓
sl	✓	✓	✓	✓	✓
sq	✓	✓	✓	✓	✓
so	✓	✓	✓	✓	✓
sv	✓	✓	✓	✓	✓
sw	✓	✓	✓	✓	✓
tl	✓	✓	✓	✓	✓
tr	✓	✓	✓	✓	✓
vi	✓	✓	✓	✓	✓
zh-cn			✓		
unknown	✓		✓	✓	✓

Table 2: Language Detection for Various URLs

References

- J. Wang, T. Kraska, M. J. Franklin, and J. Feng, "Crowder: Crowdsourcing entity resolution," *Proc. VLDB Endowment*, vol. 5, pp. 1483–1494, 2012.
- M. J. Bailey, C. Cole, M. Henderson, and C. Massey, "How well do automated linking methods perform? Lessons from US historical data," *J. Econ. Lit.*, vol. 58, pp. 997–1044, 2020.
- N. Kooli, R. Allesiardo, and E. Pigneul, "Deep learning based approach for entity resolution in databases," in *Intelligent Information and Database Systems*, pp. 3–12, Springer International Publishing, 2018.
- Y. Li, J. Li, Y. Suhara, A. Doan, and W. C. Tan, "Deep entity matching with pre-trained language models," *Proc. VLDB Endowment*, vol. 14, pp. 50–60, 2020.
- S. L. Ventura, R. Nugent, and E. R. Fuchs, "Seeing the non-stars: (Some) sources of bias in past disambiguation approaches and a new public tool leveraging labeled records," *Res. Policy*, vol. 44, pp. 1672–1701, 2015.
- G. C. Li, R. Lai, A. D'Amour, D. M. Doolin, Y. Sun, V. I. Torvik, A. Z. Yu, and L. Fleming, "Disambiguation and co-authorship networks of the U.S. patent inventor database (1975–2010)," *Res. Policy*, vol. 43, pp. 941–955, 2014.
- O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*, The MIT Press, 2006.
- K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn.*, vol. 39, pp. 103–134, 2000.
- M. D. Larsen and D. B. Rubin, "Iterative automated record linkage using mixture models," *J. Am. Stat. Assoc.*, vol. 96, pp. 32–41, 2001.
- H. Zhou, W. Huang, M. Li, and Y. Lai, "Relation-aware entity matching using sentence-BERT," *Computers, Materials & Continua*, vol. 71, no. 1, pp. 1581–1595, 2022. https://doi.org/10.32604/cmc.2022.020695
- A. Zeakis, G. Papadakis, D. Skoutas, and M. Koubarakis, "Pre-Trained Embeddings for Entity Resolution: An Experimental Analysis," *Proceedings of the VLDB Endowment*, vol. 16, pp. 2225–2238, 2023. https://doi.org/10.14778/3598581.3598594
- J. Wu, A. Sefid, A. Ge, and C. Giles, "A Supervised Learning Approach To Entity Matching Between Scholarly Big Datasets," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1–4, 2017. https://doi.org/10.1145/3148011.3154470
- Q. Wang, M. Cui, and H. Liang, "Semantic-Aware Blocking for Entity Resolution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, pp. 1–1, 2015. https://doi.org/10.1109/TKDE.2015.2468711
- O. Binette and R. C. Steorts, "(Almost) all of entity resolution," *Science advances*, vol. 8, no. 12, eabi8021, 2022. https://doi.org/10.1126/sciadv.abi8021
- Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 4, pp. 824–836, April 2020. https://doi.org/10.1109/TPAMI.2018.2889473
- G. Papadakis, D. Skoutas, E. Thanos, and T. Palpanas, "Blocking and Filtering Techniques for Entity Resolution," *ACM Computing Surveys (CSUR)*, vol. 53, pp. 1–42, 2019.
- A. Zeakis, G. Papadakis, D. Skoutas, and M. Koubarakis, "Pre-Trained Embeddings for Entity Resolution: An Experimental Analysis," *Proceedings of the VLDB Endowment*, vol. 16, no. 9, pp. 2225–2238, May 2023. https://doi.org/10.14778/3598581.3598594
- B. Genossar, A. Gal, and R. Shraga, "The Battleship Approach to the Low Resource Entity Matching Problem," *Proceedings of the ACM on Management of Data*, vol. 1, no. 4, Article 224, 25 pages, December 2023. https://doi.org/10.1145/3626711