International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-10 Issue-11, September 2021

An Effective Entity Resolution Approach for Big Data

Randa Mohamed Abd El-Ghafar, Ali Hamed El-Bastawissy, Eman S. Nasr, Mervat H. Gheith

Abstract: ***Entity Resolution (ER) is defined as the process 0f identifying records/ objects that correspond to real-world objects/ entities. To define a good ER approach, the schema of the data should be well-known. In addition, schema alignment of multiple datasets is not an easy task and may require either domain expert or ML algorithm to select which attributes to match. Schema agnostic meta-blocking tries to solve such a problem by considering each token as a blocking key regardless of the attributes it appears in. It may also be coupled with meta-blocking to reduce the number of false negatives. However, it requires the exact match of tokens which is very hard to occur in the actual datasets and it results in very low precision. To overcome such issues, we propose a novel and efficient ER approach for big data implemented in Apache Spark. The proposed approach is employed to avoid schema alignment as it treats the attributes as a bag of words and generates a set of n-grams which is transformed to vectors. The generated vectors are compared using a chosen similarity measure. The proposed approach is a generic one as it can accept all types of datasets. It consists of five consecutive sub-modules: 1) Dataset acquisition, 2) Dataset pre-processing, 3) Setting selection criteria, where all settings of the proposed approach are selected such as the used blocking key, the significant attributes, NLP techniques, ER threshold, and the used scenario of ER, 4) ER pipeline construction, and 5) Clustering where the similar records are grouped into the similar cluster. The ER pipeline could accept two types of attributes; the Weighted Attributes (WA) or the Compound Attributes (CA). In addition, it accepts all the settings selected in the fourth module. The pipeline consists of five phases. Phase 1) Generating the tokens composing the attributes. Phase 2) Generating n-grams of length n. Phase 3) Applying the hashing Text Frequency (TF) to convert each n-grams to a fixed-length feature vector. Phase 4) Applying Locality Sensitive Hashing (LSH), which maps similar input items to the same buckets with a higher probability than dissimilar input items. Phase 5) Classification of the objects to duplicates or not according to the calculated similarity between them. We introduced seven different scenarios as an input to the ER pipeline. To minimize the number of comparisons, we proposed the length filter which greatly contributes to improving the effectiveness of the proposed approach as it achieves the highest F-measure between the existing computational resources and scales well with the available working nodes. Three results have been revealed: 1) Using the CA in the different scenarios achieves better results than the single WA in terms of efficiency and effectiveness. 2) Scenario 3 and 4 Achieve the best performance time because using Soundex and Stemming contribute to reducing the performance time of the proposed approach. 3) Scenario 7 achieves the highest F-measure because by utilizing the length filter, we only compare records that are nearly within a pre-determined percentage of increase or decrease of string length. LSH is used to map the same inputs items to the buckets with a higher probability than dis-similar ones. It takes numHashTables as a parameter. Increasing the number of candidate pairs with the same numHashTables will reduce the accuracy of the model. Utilizing the length filter helps to minimize the number of candidates which in turn increases the accuracy of the approach.***

Keywords: Big data, Entity Resolution, Locality Sensitive Hashing, Hashing Text Frequency, and Blocking techniques.

I. INTRODUCTION

The representation of real-world objects is called profile or entity. Entity profile composes of a unique identifier and a set of (name, value) pairs. This model is flexible to accommodate both structured and semi-structured datasets. Two entities (e_i, e_j) are matched if they correspond to the same real-world object. Therefore, Entity Resolution (ER) could be defined as the task of finding matching entities within a set of entity profiles ϵ. When the entities arise from two diverse data sources (i.e., $\epsilon = \epsilon_1 \cup \epsilon_2$), and each source is free of duplicates, the problem is called Clean-Clean ER, otherwise, when the entities originate from the same dataset that has duplicates, the problem is called dirty ER [1]. There are many alternative names for ER such as data de-duplication, Record Linkage (RL), object matching, fuzzy matching, and duplicate detection [2]. Typical ER workflow aims to identify matched entities from one or more relational data sources with different schemas. It includes four tasks which are:

1) Pre-processing: Data pre-processing is a very critical step in ER as it highly affects the quality of the results. It involves several steps such as standardization, removing special characters, removing stop words, removing outliers, and choosing the appropriate attributes of the model. It is a time-consuming and tedious task and may be accomplished many times until acquiring an acceptable dataset suitable as an input for ER [3].
2) Blocking: Traditional ER approaches depend on applying matching techniques on a Cartesian product of n inputs entities. These approaches result in a complexity of O(n2) which causes a very high increase in the execution time for big datasets. For more efficient ER, blocking techniques have been proposed to reduce such enormous execution time.

Manuscript received on September 16, 2021.
Revised Manuscript received on September 21, 2021.
Manuscript published on September 30, 2021.
* Correspondence Author
Randa Mohamed Abd El-ghafar*, Department of Computer Science, Faculty of Graduate Studies for Statistical Research, Cairo University, Cairo, Egypt. Email: randa.mohamed@mof.gov.eg
Ali H. El-Bastawissy, Faculty of Computer Science, Modern Sciences and Arts University, Cairo, Egypt. Email: aelbastawissy@msa.eun.eg
Eman S. Nasr, Independent Researcher, Cairo, Egypt. Email: nasr.eman.s@gmail.com
Mervat H. Gheith, Department of Computer Science, Faculty of Graduate Studies for Statistical Research, Cairo University, Cairo, Egypt. Email: Mervat_gheith@yahoo.com

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

Blocking techniques depend on distributing the big dataset into several small blocks such that elements that reside in the same block are more likely to be matched. Many blocking techniques have been used in literature as clarified in [4].

3) Similarity Computation: At each block, candidate entities are compared using one or more similarity functions to compute the similarity between them. Many similarity functions could be utilized as described in [5], [6].
4) Matching and Classification: The computed similarities will be used to decide the matching status of the objects. The decision could be taken based on a similarity threshold, rules, weighted similarity values, or it can be derived from a Machine Learning (ML) classifier that has been trained with suitable data.

New programming models as Hadoop MapReduce (MR) and Apache Spark are used to run complex jobs in parallel to increase the response time. They can utilize commodity hardware and provide fault tolerance, high scalability, and flexibility to distributed processing. MR is an early popular framework because of its simplicity and easiest of programming however, it has several limitations [4]. In addition, every single job has to read the input data from Hadoop Distributed File System (HDFS), process it, and write it back to HDFS. Iterative applications such as machine learning and graph analytics require multiple access to the same dataset, which leads to additional overhead. To overcome such limitations, Apache Spark is used. Apache Spark is a clustering framework that utilizes memory with different APIs in Scala, Python, and Java. Apache Spark emerged and was boosted by its Resilient Distributed Dataset (RDD), which supports in-memory processing capabilities that are used to cache intermediate data access of subsequent tasks accomplished by the worker nodes. Since RDD is retained in memory, different tasks can iterate over it multiple times efficiently [7].

One popular algorithm of Fuzzy or approximate entity matching is edit distance, which is the number of inserts, delete, and substitute that is used to transform one string into another. However, this algorithm is too computationally expensive for large datasets. To define a good ER approach, the schema of the dataset should be known, however, schema alignment of a different dataset is a heavy task and requires the help of a domain expert. Schema-aware techniques have two main issues especially when concerning with big data: 1) it requires schema alignment, which may be difficult to be achieved with big data, 2) it requires either classification algorithms, labeled data, or domain expert to select which attributes to match [8]. Schema-agnostic-meta blocking tries to mitigate such issues as in [8], [9] by considering each token in the record as a blocking key regardless of the attribute in which it appears. Schema agnostic is coupled with meta-blocking to minimize the number of false negatives. Meta blocking aims to re-structure the blocking collection by removing the slightly promising comparisons. However, it requires the exact match of tokens, which is very hard to occur in real-world datasets, and it results in very low precision.

To overcome such issues, we proposed an effective and approximate ER approach for big data implemented in Apache Spark. The proposed approach could be employed to avoid schema alignment because it treats the attributes as a bag of words and a set of N-grams. Utilizing Apache Spark helps to address many limitations of using the MapReduce frameworks such as the data skew problem that happens due to the unequal block sizes and results in unbalanced entity pairs which in turn lead to severe imbalances in the reduce phase. Our goals are: (1) accomplishing an increase in the F-measure. (2) minimizing the execution time accomplished during all ER tasks by all worker nodes to detect as many potential duplicates as possible (i.e. Given a dataset, classifier, and its input). The contributions of this paper could be summarized as:

1) It is a generic Approach that has the following advantages:
   - It describes the individual processing steps that form our proposed generic ER approach.
   - It can classify pairs of records written in different languages.
   - It can accept diverse types of datasets (structured, semi-structured, and unstructured) and use them efficiently.
   - It supports parallel processing by distributing the big dataset to several working nodes and uses different levels of parallelization to examine the running time of the algorithm. It can distribute the similarity calculation and classification between the available computational working nodes and scale well with the available computational resources. Classic ER focus on effectiveness. As far as we know, the research of parallel and scalable ER is still not much.
2) Proposing an end-to-end ER approach using several NLP techniques such as removing stopwords, Soundex, and Stemming to pre-process the record pairs before classify them using the proposed approach. Using NLP techniques contributes to minimizing the length of the attributes, therefore; it helps to enhance the performance of the proposed approach. Using the phonetic codes reduces the problem of approximate matching of unclean data to a problem of exact matching of phonetic codes, which in turn leads to faster processing.
3) Using feature re-engineering to create the CA participates in achieving better performance compared to using separate WA.
4) Applying MinHash Locality Sensitive Hashing (LSH) that maps the similar inputs to the same hash code with a greater probability than dissimilar ones. It is used to create an indexing structure as they extract the short signature of the objects using hashing functions. LSH maintains the object in buckets according to their signature. Using hashing function is more efficient as it takes less memory and helps to reduce the performance time. As LSH uses hashing to conduct an efficient and rapid search, it was used through our proposed ER pipeline. Using Stemming and Soundex led to a decrease in the features' size therefore, they result in an improvement of the performance of the proposed approach.

Retrieval Number: 100.1/ijitee.K950309101121
DOI: 10.35940/ijitee.K9503.09101121
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

5) It utilizes the semantic associations or relationships between attributes to predict their missing values. Including the association among the attributes significantly enhance the overall accuracy of the proposed approach.
6) Proposing seven different scenarios using different NLP techniques and evaluating them to choose the most efficient and effective one.

II. RELATED WORK

If the data being investigating is big data, more challenges are added to the traditional ER. Such as the huge volume of data, the diversity of sources, the variety of structure types, the high rate of data changing, and the quality of the data. Therefore, Traditional ER approaches become inefficient and ineffective to deal with such increasing size and diverse structure. Two solutions could be used to enhance the efficiency of ER. The first one is the use of parallel processing approaches and the second one is the use of blocking and pruning techniques. Apache spark is a framework used for distributed big data processing that supports in-memory computations. Several researchers adopt Apache Spark-based approach to solve the problem of ER. Pita et al. [10] and Chen et al. [11] proposed Spark-based frameworks that consist of four phases; 1) Data quality assessment, 2) Data pre-Processing, 3) Performing Record Linkage, and 4) Evaluation of the approach. A scalable duplicate detection workflow using Apache Spark has been presented to solve the problem of duplicates in Adverse Drug Reaction (ADR) report databases which come from a variety of sources [12]. To reduce the high computational cost of K-Nearest Neighborhood (KNN), they proposed a method that uses a KNN classifier to identify labeled report pairs. The wrong choice of window size of KNN has highly affected the performance of the proposed approach. For parallel computing, they partitioned the labeled data into clusters. Mestre et al. [7] proposed Spark Duplicate Count Strategy (SDCS ++), a Spark-based approach for adaptive Sorted Neighborhood Method (SNM). The proposed approach considered load balancing techniques to solve the data skew problem by applying a strategy for automatic data partitioning with a load balancing between all the worker nodes. Gatius et al. [13] used Apache Spark to resolve duplicates in a scholarly dataset. First, the used dataset is divided into 3 parts; the first one is about the article, the second one is about the journal, and the last one is about the article in the journal. Second, each part is compared separately to decide whether parts refer to the same article or not. Two articles are marked as duplicates if three parts agree that they are referring to the same article. The final similarity between two records of articles is calculated by comparing fields of two articles and each of them is given a weight according to their importance. LSH is used to construct the blocks. To test the proposed solution, they manually labeled the records of the dataset as duplicate or not which is a tidy process, especially with very large datasets. In addition, the used dataset is very small as it is only having 686 records which may not be adequate to test the solution. Almmi et al. [14] proposed a parallel algorithm utilizing Apache Spark to partition the dataset using K-Medoid algorithm. As the k-means algorithm clusters the records into k clusters such that the intra-cluster similarity is high whereas the inter-cluster similarity is low, K-Mediod solves the same problem but with great complexity. They choose K-Mediod to keep a good quality of the problem of deduplication. To resolve the high computational time, they used Apache Spark to parallelize the algorithm but there was a remarkable unbalance between the different nodes. Martinez et al. [15] proposed a methodology for finding the best linkage between elements of two different trajectories sources, namely radar and GPS (Correlation Position Reports (CPRs) and ADS-B captured positions). The methodology uses a number of possible identifiers, gives weight to each one based on its capacity to correctly identify a match or non-match, then these weights are used to compute the probability that two records from diverse data sources correspond to the same entity. They used Apache Spark to perform an inner join using date and call signs identifier. No blocking or parallel distribution is used and no pruning strategy is exploited in the proposed approaches presented in [10], [11], [12], [7], [13], [14], [15]. For handling the diverse structure of the big datasets, Gagliardelli et al. [16] introduced SparkER which is a distributed Entity Resolution (ER) tool that can run on top of Apache Spark. In the first version of SparkER, their focus was on the blocking step and they implemented both structure-agnostic and blast meta-blocking approaches. Entity matching and entity clustering modules have been extended to SparkER. The main objective of meta-blocking is to remove the least promising comparisons. Gagliardelli et al. [17] present BigDedup, an efficient toolkit for duplicate records detection of big datasets using Apache Spark. BigDedup can process both structured and unstructured datasets efficiently. Two blocking techniques have been implemented to decrease the number of comparisons. The first one is schema-agnostic, which does not consider the schema. Schema-agnostic is a heavy task especially in the case of different data sources and results in a high recall but very low precision. To overcome such a problem, Simonini et al. [9] proposed Loose Schema-aware Blocking (LSB), a schema-aware version of token blocking. LSB applies Locality Sensitive Hashing (LSH) on their values and gives them a weight to generate clusters of similar attributes then, token blocking is applied to cluster together records that have the same blocking key. However, using blocking is not enough to reduce the number of comparisons, therefore, more sophisticated approaches are required. Papadakis et al. [18] proposed a meta-blocking technique to solve that problem. It depends on creating clusters of entities sharing the same tokens, and then construct a Graph such that nodes denote entities and edges represent the weights of the shared tokens. All edges that have a weight less than a certain threshold are pruned. BigDedup is only evaluated using a single node. They only used three datasets with a very small size, which doesn't require a cluster to perform the proposed algorithm. Papadakis et al. [19] present the Java gEneric DAta Integration (JedAI) system, an open-source ER system. JedAI develops series of state-of-the-art ER approaches that have been suggested and examined. It offers many characteristics: 1) It permits for building millions of ER pipelines, 2) It applies combinations of structured and/or semi-structured data, 3) It can run on both stand-alone or cluster systems through a parallel implementation using Apache Spark, 4) It supports both budget-agnostic and schema-agnostic for batch ER.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication
© Copyright: All rights reserved.

Sagi et al. [20] present a multi-source uncertain ER model. The used dataset massively has multiple sources and requires multi-level ER. The proposed model used MFIBlocks algorithm and a machine learning approach based on Alternating Decision Tree (ADT) to transform soft clusters into ranked clusters of records. MFIBlocks allows an automatic, dynamic, and flexible choice of blocking key so that different blocks could be created based on different keys. No pruning techniques are used. Apache Spark is only used in a local mode as no parallel distribution is performed.

The goal of Privacy-preserving record linkage (PPRL) is to links entities across multiple databases without revealing any sensitive information about these entities. Different PPRL approaches have been proposed to conduct PPRL by encoding sensitive values. Karakasidis et al. [21] proposed a parallel protocol for Privacy-Preserving Record Linkage (PPRL) based on phonetic encoding using Apache Spark. The proposed approach combines both Soundex and New York State Identification and Intelligence System phonetic encoding algorithm (NYSIIS) algorithms. As Soundex is vulnerable to errors that happen at the prefix of the encoded text, the proposed protocol deploys an optimization to the algorithm by encoding the reverse of the original text with the second phonetic algorithm. Brown et al. [22] presented a new hybrid cloud model for PPRL. They used containers to distribute the record linkage workload across multiple nodes. A proprietary data linkage management system called LinXmart was used. Three synthetic datasets were generated to be used in the RL process. All step function tasks used on-demand resource provisioning for computation using a cluster managed by Amazon Web Service (AWS). The first task splits the file into many bins of approximately equal comparison space. The second task runs a node array batch job to de-duplicate each bin independently. The third step function task classifies the new pairs by using AWS Glue to make them available for use by other AWS analytical services. Pita et al. [23] present AtyImo, a scalable tool that links very large datasets and variable data quality issues. It is a modular pipeline that has encapsulated components for data pre-processing, pairwise comparison, and matching decisions. For Anonymization, AtyImo uses Bloom filter as health data is very critical. For pairwise comparison, AtyImo proposes two approaches. The first is full probabilistic where the Bloom filter representing linkage attributes is completely compared. The second one is a hybrid approach of deterministic and probabilistic rules that are applied to the linkage attributes. A manual review over samples of databases has been performed to be used as golden standards. The evaluation of AtyImo's scalability and accuracy depends only on the small size. AtyImo is implemented over Apache Spark. No blocking or pruning techniques are implemented in [21], [22], [23] except for the last one as Different predicts have been analysed for blocking selection. Chen et al. [24] examine the use of Spark-SQL for efficient parallel entity resolution. They apply different persist() options then evaluate the efficiency of the proposed algorithm. They concluded that the performance of Spark-SQL ER is 3 times better performance when employing the best persistence options. They implement the same ER algorithm using Spark-RDD-based and compare it with Spark-SQL-based. They found that Spark-SQL-based achieved 2.2 times better efficiency than Spark-RDD-based ER. The process begins by loading the data in Spark dataframe with a load balancing distribution to all available nodes. Then the pre-processing and the blocking phases are performed to standardize and blocking the dataset. Next, the similarity between the candidate pairs is calculated. No pruning techniques have been used to eliminate unnecessary comparisons.

III. THE PROPOSED ENTITY RESOLUTION APPROACH

This section presents our proposed ER approach that can accept different types of datasets. The workflow of the proposed ER approach is illustrated in fig.1. The workflow consists of five modules. 1) In the first module, the dataset is acquired. The proposed approach can accept different types of datasets such as structured data extracted from relational datasets, semi-structured data as XML and JSON files that could be scraped from social media, and finally unstructured data such as free text files. Multiple sources could be used to scrape or download the required dataset to be used during the matching process. 2) In the second module, the feature selection is performed to only select the features that will participate in the ER process. Next, the data pre-processing is executed to guarantee an acceptable quality of the dataset involved in all subsequent phases of the matching process. 3) In the third module, the setting of all criteria involved in all the phases of the proposed approach is performed. Multiple selection criteria are set such as the Significant Attributes (SA) selection, the BLocking Key construction (BLK), scenario selection, the value of N-grams, and finally the threshold value that will be used to detect duplicate objects. The setting selection criteria could be done with the help of an expert, Machine Learning (ML), or the operator of the proposed approach. 4) In the fourth module, the construction of the ER pipeline module is performed. The pipeline takes as an input the selection criteria determined in the last phase. The pipeline consists of five subsequent stages; the first stage is the generation of tokens using Regular Expression (RE). The second stage is the generation of n-grams, which is the process of representing the string as a set of substrings of length n called n-grams. In the third stage, the hashing TF is applied to convert each n-grams to a fixed-length feature vector by applying a hash function. In the fourth stage, we applied MinHash Locality Sensitive Hashing (LSH) that maps similar data inputs to the same hash code with a greater probability than dissimilar items. In the last stage, the similarity between vectors is measured using a similarity measure to classify the records to matched or un-matched records based on a calculated threshold value, ML, or rule-based. The output of the pipeline is a dataset of duplicate objects. 5) In the last module, the clustering of similar objects is done such that similar records will reside in the same cluster. To cluster the objects, multiple approaches could be utilized such as ML-based, Graph based, and in-house development-based. The next subsections will explain these five modules in detail.

Retrieval Number: 100.1/ijitee.K950309101121
DOI: 10.35940/ijitee.K9503.09101121
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

A. *Dataset Acquisition Module*

To build any proposed approach or develop ML model, the relevant dataset must first be acquired. This dataset consists of data gathered from different and diverse sources then, they are combined together in a suitable format to create the final dataset. Dataset format may vary according to use cases. Multiple sources could be used for acquiring data. There are multiple online sources to download the dataset such as Kaggle. Another alternative is to contact the dataset providers to obtain it or to buy it if it was not publically available. Sometimes historical local government data could be used but in most cases, it is not allowed for security purposes. In some cases, we may need to scrap the required data and use it. Different APIs could be used to extract the data and create a dataset. Dataset comes from diverse sources and has many formats: structured, semi-structured, and non-structured, thus making the data more complex [25]. Only 10% of all these data is structured data compared to 80% that is unstructured data [26]. The proposed approach can accept all types of the dataset (structured, semi-structured, or unstructured). For the unstructured or semi-structured data, the extracted datasets could be re-constructed on the fly in a form ready for analytics purposes. For example, Twitter API could be used to scrape tweets about a certain topic. Tweets are written in an unstructured format and stored in a JSON file format. The extracted JSON file could be transformed into dataframe, therefore; multiple datasets could be built on the fly and many types of analytics could be accomplished on them.

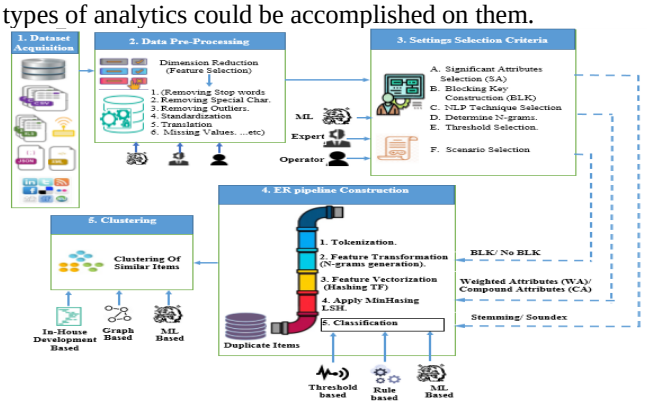


Fig.1 The Workflow for the proposed ER approach

B. *Data Pre-Processing Module*

After the acquisition of the dataset that will be used in the ER process, the dataset is transformed from a high dimensional space to a low dimensional space so that the low dimensional space keeps the significant properties of the original dataset. Such transformation is called dimensional reduction [27]. Feature Selection could be defined as the process of automatically or manually select the relevant features, which contribute most to the ER problem and leave all the rest of the features which will not be used. Feature selection aims to remove non-informative or redundant predictors from the proposed model [28]. Feature selection helps to reduce the computational cost of ER approach and improve the performance time [28]. To choose the relevant features, many alternatives could be used such as an operator, an expert, or a ML model. The operator uses his experience to select the most relevant features, which will perform better in ER problems. An Expert is a person with special knowledge or skills in a particular domain or area. Two types of ML models could be used, supervised and unsupervised. Supervised ML model uses a target feature such as methods that remove irrelevant features. In an unsupervised ML model, there is no target feature such as methods that remove redundant features using correlations [28]. A survey of dimension reduction techniques could be found in [27]. The selected features maybe not be clean enough to be processed and used in the subsequent phases. Data Quality is concerned with fitting the data for use by consumers that interpret, access, analyze, and use the data during their working activity [29]. Accuracy, completeness, and consistency are three dimensions that could affect the quality of the data [30] [31]. To guarantee high-quality data, it is important to preprocess it. Regarding the fact that high-quality data leads to better models and predictions, data preprocessing is an essential phase in the data science, ML, and AI models. Data pre-processing is the process of converting raw data to formatted data that improve the data matching results to be more precise and efficient [32] [33]. Data pre-processing involves cleaning and standardizing the dataset. It is very important to identify how to correctly deal with the missing values of the attributes. Failure to do it may draw inaccurate conclusions and decisions. There are many ways to handle missing data such as calculating the mean and deleting the whole row [34]. For accomplishing successful data preprocessing, it is vital to have an overall picture of the data. Descriptive data summarization techniques could be used to detect the properties of your data and discover which data values should be treated as outliers or noise. Descriptive analysis characterizes a certain phenomenon. It is used to answer enquiries about who, what, where, when, and to what extent. It aims to recognize and describe trends and variation in populations, describe samples in studies to identify the causal effects, and make new measures of key phenomena [35]. The output of data pre-processing module is a standard format of all attributes involved in ER process.

C. *Setting Selection Criteria Module (Configuration Module).*

It is the third phase of the proposed approach. It accepts the pre-processed dataset results from the last module. In this module, the setting criteria are determined and will be used in all the subsequent modules. Many setting criteria are selected in this module. The first one is the Significant Attributes (SA). SA are the attributes that will be used during the ER process. The second one is the BLocking Key (BLK) construction. In the blocking key construction, the attributes that compose the blocking key are selected and the blocking key method is determined. ER is only performed between records located in the same block to enhance efficiency and accuracy [36] [37]. A taxonomy of the blocking dimension could be found in [38] [39]. The problem owner, an expert, or ML model could be used to determine the SA. The SA will compose two types of attributes: the first one is the Compound Attributes (CA), while the second one is the Weighted Attributes (WA).

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication
© Copyright: All rights reserved.

The CA is composed of the concatenation of more than SA that will be involved in the ER process. The WA are the same attributes that compose the CA and each of them is tested separately with a certain weight given to each attribute. Then, these weights will be used during the calculation of the final similarity. The third one is determining the N-gram that will be used in the pipeline module. The fourth one is the determination of the threshold value. The threshold value is used to classify candidate records as duplicate or not. Candidate Records are said to be duplicates if they have a value of similarity greater than or equal to a pre-defined threshold value, otherwise, they are not considering duplicates. Given a similarity function f, an entity e, a collection of entities E, and a similarity threshold Θ, A threshold-based Similarity search is identified by all entities e' ∈ E having similarity to e at least Θ, i.e., f (e, e') ≥ Θ [40]. Finally, the ER scenario is selected. We proposed seven different scenarios, which are:

1) Scenario 1 (S1): Using the WA and removing stopwords.
2) Scenario 2 (S2): Applying Stemming techniques to the CA and removing stop words.
3) Scenario 3 (S3): Applying Soundex techniques to the CA and removing stop words.
4) Scenario 4 (S4): Appling Stemming and Soundex techniques to the CA and removing stop words.
5) Scenario 5 (S5): Applying neither Stemming nor Soundex techniques to the CA and do not remove stopwords.
6) Scenario 6 (S6): Applying neither Stemming nor Soundex techniques to the CA, don't remove stopwords and don't apply blocking key.
7) Scenario 7 (S7): Applying length Filter to scenario 4.

In the proposed scenarios, we applied different NLP techniques such as Stemming and Soundex to test their effect on the performance of the proposed approach. Stemming is the process of transforming the word to its root representation. Soundex is a phonetic algorithm for indexing names by sound [21]. In the first scenario, WA is used as an input to the ER pipeline after removing the stopwords from them. In the second scenario, we applied the Stemming technique to the CA and remove the stopwords from them. In the third one, we applied Soundex techniques to the CA after removing stopwords from them. In the fourth scenario, we applied both Stemming and Soundex techniques to the CA and remove the stopwords from them. In the fifth one, we applied neither Stemming nor Soundex techniques to the CA without removing stopwords from them. In the sixth scenario, we do not apply any Soundex, Stemming, nor blocking techniques. In the last scenario, we proposed a length filter and applied it to scenario four. The length filter is used to only compare the CA that reside in nearly the same length of vectors. All the selected setting criteria will be used in the next module described in the next sub-section.

D. Entity Resolution Pipeline Module

In this section, the ER pipeline takes as input all the setting criteria selected in the last module. The ER pipeline consists of five stages, which are used to transform the input string into its numeric representation and apply the similarity algorithm to them. The first stage is tokenization, which is the process of taking the input string and breaking it into its individual terms that are usually words. In the second stage, we generated n-grams, which is the process of representing the string as a set of substrings of length n called n-grams. Producing n-grams is an efficient way to generate the numerical representation of the string. For example, bi-grams, and tri-grams representations of the word "English "are {En, ng, gl, li, is, sh}, and {Eng, ngl, gli, lis, ish}. In the third stage, we applied Hashing TF. After converting the input string to its equivalent string of n-grams, then, each of the n-grams is converted to a fixed-length feature vector by applying a hash function to each of them and using the hash function as an index directly into the array. In the fourth stage, we apply Min Hashing Locality Sensitive Hashing (LSH). After converting n-grams to feature vectors, the next goal is to replace potential large sets with a small representation that is comparable for two strings and estimate the similarity between the two strings. Locality Sensitive Hashing (LSH) aims to find similar pairs in a very large collection of items without looking for every pair to avoid a quadratic process of comparisons that results in insufficient memory. LSH maps similar input items to the same hash code with greater probability than dissimilar ones [41]. It is a generalized idea of hashing items into bins many times and looking for items that exist in the same bin at least one such that only high similarity items are likely to fall into the same bucket. The dataset may be blocked according to the selected scenario. After blocking the dataset to multiple chunks and transforming all features into their numeric representation, the similarity between candidate records is compared. Approximate similarity join is performed using a similarity measure. Finally, in the last phase, the two input items are classified as duplicates pairs or not. The decision could be taken depending on a threshold value-based as in [21] [14] [24] [17] [16], rule-based as in [43], [44], [45] or ML-based as in [46] [45] [2] [47] [48]. The output of the ER pipeline is a dataset of duplicate items.

E. Clustering Module

Clustering is defined as the process of identifying groups or clusters within multi-dimensional data depending on a similarity measure [49]. In the context of ER, clustering is the process that aims to group similar objects together into the same cluster. Clustering targets to reach a consistent decision of partitioning the dataset such that each partition denotes a distinct entity [50]. Multiple approaches could be used to group similar entities to the same cluster. The first one is using transitive closure property. Let A, B, and C are three records, if A is matched with B and B is matched with C, then A is matched with C. The transitive closure property has some advantages regarding other transitive approximations, but on the other hand, suffers from the chaining effect where two objects unrelated by the original relation can become very related by the transitive relation [51]. The second one is using graph-based clustering. A graph G = (V, E) consists of a set of vertices (V) or nodes and a set of Edges (E). Nodes are connected using edges, which represent the relationships between nodes. Regarding the ER, Nodes indicate the objects, or records to be matched and Edges specify the similarity between these nodes.

Retrieval Number: 100.1/ijitee.K950309101121
DOI: 10.35940/ijitee.K9503.09101121
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

Nodes located in the same cluster denote the same entity and refer to a completely different entity with nodes outside that cluster [52]. A cluster is defined as a group of nodes, which refer to the same entity in the same real-world [52]. One of the simplest Graph strategies is to efficiently cluster graphs into connected components by a single scan of edges in the graph. Many ER approaches utilize Graphs for clustering similar objects could be found in [53], [54], [55], [56], [57], [58], and [59]. The third one is using ML clustering approaches. ER could be considered as a classification problem. For all entity pairs p ϵ R x S of two datasets R and S, a classifier is used to determines if the entity pairs are matched or non-matched. The current state of arts exploits supervised and unsupervised machine learning approaches in ER problems. Supervised learning approaches aim to classify pairs of records as matched and unmatched based on a labeled dataset that will be trained using a machine learning classification model. In unsupervised learning-based approaches, no labeled data is available. Supervised machine learning techniques require manual labeling of the dataset, which is hard to achieve especially with a large dataset. The similarity values of the candidate pairs serve as the features for the classification [2]. The Supervised learning-based ER workflow is composed of two phases; training and application (testing). The training phase involves a training dataset where the entity pairs are labeled as matched or non-matched. Each row of the training data is composed of similarity values between two pairs calculated by different matches. The pairwise similarity values serve as the features of the classifier. In the application phase, new unlabeled data is verified to be labeled by the classifier and test its accuracy. Ektefa et al. [60] present a comparative study of classification techniques for an unsupervised record linkage model. They used Support Vector Machine (SVM), Nieve bays, decision tree, and Bayesian network to train and build the models, and then, the testing set is classified using the training set. Multiple unsupervised ML approaches could be found in [61], [62], and [63].

IV. EXPERIMENTAL RESULTS

In this section, we will discuss the experimental results of the proposed approach. The implementation is done using Apache Spark 2.4.3 installed on Windows 7 operating system. To enhance the performance of the proposed ER approach, we used spark dataframe, as it is faster than the regular RDD. In addition, we used persist() which is an optimization mechanism used to store the intermediate computation of an RDD to be reused in subsequent actions. Using persist allows each worker node to store its partitioned data in memory or disk and reuses them in other further actions on that RDD. We run the experiments on a local PC that has Intel ® core ™ i7.35174 CPU @ 1.90 GHZ 2.4 GHZ, and 16 GB RAM. Different levels of parallelization have been implemented to examine the running time of the proposed approach. Apache Spark was running on windows 10. We developed five servers in the cluster; one of them is the driver node and the others are the working nodes.

For testing the effectiveness of the prosed ER approaches, a labeled dataset with known duplicate records is required. We surfed the internet for a synthetic dataset with known labeled duplicate records to use for testing our model. We found several synthetic duplicate datasets but unfortunately, they may need many manual labeling to be more accurate. Since many true duplicated are not mentioned as real duplicates and many records that are labeled as duplicated are not truly duplicated with a good percentage as we found many records labeled as duplicated with a similarity percentage less than 40%. In addition, many datasets had a very small size, which will not be appropriate for our proposed model. These reasons motivated us to develop a DupGen tool [38] that accepts a CSV dataset as an input and generates synthetic records according to certain criteria such as removing, swapping, and inserting char/word or more as depicted in Fig.2. The generated dataset will be used to test the effectiveness of the proposed approach while the efficiency is measured by the achieved processing time. The proposed approach can accept different structures of the dataset but to evaluate our approach, we used a synthetic dataset generated by our developed tool DupGen. As per our knowledge, no ER approaches for big data deal with unstructured datasets but our approach can use the three types of datasets effectively. The original dataset is about music and consists of 11 original attributes. To measure the performance of our proposed model, we generated four additional attributes, which are OriginalID, IsOriginal, No_Of_Changes, and Changes Description [38]. The original attributes are TID, CTID, Source ID, ID, number, length, title, artist, album, year, and language. To reduce the search space, we drop the attributes TID, CTID, Source ID, ID, number, and length, which will not be used during the ER process. To test the scalability of the proposed model, we generated many synthetic datasets of different sizes which are 25,000, 50,000, 100,000, 250,000, 500,000, 750,000, and 1,000,000 records. The duplicate records are

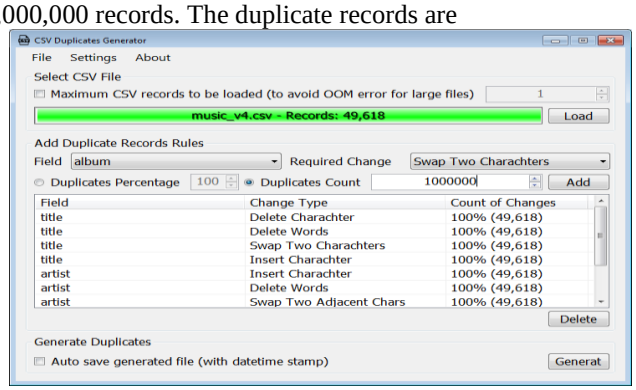


Fig.2 DupGen Tool [38]

Distinguished by checking the similarity of the corresponding records using several selected attributes. These attributes include title, artist, album, year, and language. The values of the similarities will compose the similarity vector. To evaluate the proposed algorithm, we used OriginalID, which is shared by all duplicate records.After generating the dataset, the data-cleaning phase is started. Data cleaning is a very tiring and time-consuming task. It is a very important task because it highly affects the quality of the ER results due to the concept of garbage in garbage out. Data cleansing is done gradually in many phases.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication
© Copyright: All rights reserved.

To enhance the quality of ER, we used many NLP techniques. First, we remove all special char, punctuations, extra spaces from all fields involved in the ER process. Then, we remove non-numeric char from the year field and numeric char from the rest of the fields. We remove outliers from year and language fields such as removing negative values of year or values greater than four digits. The Regular Expression (RE) library has been used to find and remove unwanted patterns. The missing values of the dataset attributes are a big problem since they will result in inaccurate results. Many scenarios have been done to solve the problem of the missing values of title and languages such as:

1) Predicting the values of language by utilizing the relationships between the attributes. As an example, by using common words of a certain language, we can predict the language. For example, if the title or the album has any of the words ("the", "and", "you", and "is") then, we can conclude that the language is English. We used title and album to prepare a list of the most frequent words for each language then, we remove the common words between languages, for example, we removed from the list 'de', 'la' between French and Spanish. Finally, we used the remaining list to predict the null values of the language.
2) Removing the whole record if it has no value in title, artist, album, year, or language.

In addition, many filtrations to the records have been done. First, we filter the records to remove the records that have "unknown "value in the title attribute. Second, we removed all the records that have a length of characters in the title attribute of less than two characters. After trying to fill the null values of the language attribute, we remove all the records that have null values of title and artist attributes. The output of this phase is a standard format of all attributes involved in ER to be ready for the further modules of the proposed approach. After pre-processing the dataset, the next step is to choose the setting selection criteria. during this phase we choose the SA, whether we will apply blocking key or not, NLP techniques (Stemming, Soundex, both, none of them), the threshold value of ER, and finally the used scenario. We generate n-grams of length 3 characters. To classify the records, we used a threshold value of 70%. For blocking, we used the first three letters of the language and the last two letters of the year as a blocking key. All these settings will be used as an input to the next phase (i.e. ER pipeline). The pipeline can take two types of attributes according to the used scenario; the first one is the Compound Attribute (CA) while the second is the Weighted Attributes (WA). For the CA, we choose title, artist, and actor as compound attributes. For the WA, we used the same three attributes (title, actor, and album) with the weights (0.6, 0.2, 0.2) as the relative importance of title as an identifier of the song is higher than both actor and album. These two inputs are used in the seven proposed scenarios which will be evaluated in terms of efficiency and effectiveness. The pipeline goes through five stages. In the first one, we generate the tokens from the CA or WA. We used Spark RegexTokenizer to transform the input string to tokens. In the second one, we create n-grams of length 3 characters. We used Spark n-grams feature transformer to convert the compound string into an array of n-grams. In the third one, we used HashingTF to create the feature vectors of fixed length. In the fourth one, we apply MinHashLSH to map similar inputs to the same hash code with a higher probability than dissimilar items. Finally, in the last stage, the candidate objects are classified to duplicate or not depending on a similarity method. To measure the similarity between the candidate pairs, several similarity measures could be utilized such as Jaccard Distance, and Euclidean Distance. We used Jaccard similarity measure, which is also known as Jaccard similarity index. It is the number of the shared objects (bi-gram in our case) divided by the total number of objects. The output of this phase is a dataset of all the duplicate objects. The final phase of the proposed approach aims to cluster the objects such that all entities that represent the same real-world object are residing in the same cluster. Many duplicate records could be missed due to many reasons such as the unsuitable selection of the blocking key, and wrong selection of the threshold value; therefore, we apply transitive closure to cluster similar records together and detect as many duplicate pairs as possible. Fig.3 illustrates the ER algorithm for big data. To reduce the performance time, we also provide a length filter, which is used to filter the vectors that will be compared. If the vectors pass the length filter test, they will be compared otherwise they will be excluded from the comparison process. To pass the length filter test, vectors should be near of the same length i.e. if X and Y are two vectors, they will only be compared if their length is exceeded or decreased by a certain percentage otherwise, they would not be compared. The vector length filter is used in the last scenario. For example, if we apply a 20% length filter of the vectors, it means that we will

Input: pre-processed dataframe (df), Compound Attribute (CmdAtt), number of ngrams (num), numHashTables=nht
Output: matched dataframe (matched_df)

```
Entity_Matching_Model (df) {
        Generate_BlockingKey(DF){
          Return (substring(DF.attibute1, 0, 3) + substring(DF.attibute2, 2, 2)) }
        //Create blocking key column
        df.withColumn('blocking_key', Generate_BlockingKey(df))
        //Create a Pipeline to construct the ML Model
  Pipeline = pipeline (Stages =RegTokenizer (inputCol = CmdAtt, outputcol = tokens),
                Ngrams= (n=num, inputCol = tokens, outputcol = ngrams),
                HashingTF (n=num, inputCol = ngrams, outputcol = vectors),
                MinHashLSH ((inputCol = vectors, outputcol = LSH, numHashTables=nht))
  // Create, fitting, and transforming the model
        // Create hashed dataframe.
  Model = Pipeline.fit(df)
  df_hashed = model.transform(df)
        // Self join hashed_df as (first_ hashed_df and sec_hashed_df) and calculate the similarity.
        If (first_ hashed_df.bloking_key == sec_ hashed_df.bloking_key ) then
            matched_df = model.stages
```


Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication
© Copyright: All rights reserved.

[-1].approxSimilarityJoin (df_hashed as , df_hashed, 0.20, distCol ="JaccardDistance")
end if
Return (matched_df)}

Fig.3 The Proposed Entity Resolution Algorithm for Big Data.

only compare vectors that are decrease or increase by each other by only 20% and vectors located outside such boundaries will not be compared. We evaluate the seven proposed scenarios using the confusion matrix. To evaluate the proposed approach, we used the golden standard dataset generated by our developed tool DupGen. To evaluate the efficiency, we measure the entire execution time of the whole phases involved in the ER process. To evaluate the effectiveness of the proposed ER model, we applied matrices (1), (2), and (3) to each of the above input scenarios [64].

1) Recall: It is used to measure the percentage of matches that are discovered by the classifier. It is calculated as:

$$\frac{TruePositive(TP)}{TruePositive(TP)+FalseNegative(FN)} \quad (1)$$

2) Precision: It measures the percentage of the comparisons that result in matches. It is calculated as:

$$\frac{TruePositive(TP)}{TruePositive(TP)+FalsePositive(FP)} \quad (2)$$

3) F-measure: It is the harmonic mean of precision and recall. It is calculated as:

$$\frac{2\times \Pr ecision\times \operatorname{Re} call}{\Pr ecision+\operatorname{Re} call} \quad (3)$$

Table 3 illustrates the achieved performance time, and f-measure for all the seven scenarios using the different size of the dataset, which are 25,000 record, 50,000 record, 75,000 record, 100,000 record, 250,000 record, 500,000 record, 750,000 record, and 1,000,000 record. For 500,000, 750,000, and 1,000,000 records, we distribute the records to two, three, and four working nodes respectively. Several iterations have been made to equally distribute the dataset between the working nodes. To avoid the problem of data skew and improve the load balancing, the entities have been divided between worker nodes in an almost equal manner. To measure the performance time in the case of distributing the dataset to more than a working node, we took the highest nodes' performance time. Table 1 shows the achieved performance time (P.Time) and F-measure (FM) in case of single node (25,000 records, 50,000 records, 75,000 records, 100,000 records, 250,000 records) and multi-nodes (500,000 records, 750,000 records, 1,000,000 records) for each of the proposed seven scenarios (i.e. Sc1, Sc2, Sc3, Sc4, Sc5, Sc6, Sc7). Therefore, the results could be summarized as:

A. Implementing the proposed seven scenarios using a single node.

We used Apache Spark installed on a single node to implement the proposed seven scenarios of the datasets

Table- I: The achieved effectiveness and efficiency in the seven different scenarios using different dataset size

25,000 Record							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time	00:02:45	00:01:53	00:01:47	00:00:30	00:00:54	00:04:19	00:00:40
FM	86%	95%	92%	91%	99%	98%	91%
50,000 Record							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time	00:03:55	00:02:49	00:02:12	00:02:11	00:02:40	00:24:43	00:03:37
FM	80%	96%	92%	90%	99%	99%	90%
75,000 Record							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time	00:05:59	00:03:30	00:03:23	00:03:40	00:04:35	01:07:21	00:05:04
FM	89%	96%	93%	91%	99%	99%	91%
100,000 Record							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time	00:16:32	00:07:34	00:07:04	00:05:43	00:07:04	00:40:09	00:07:46
FM	90%	97%	94%	92%	99%	99%	92%
250,000 Record							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time	00:20:03	00:10:01	00:10:00	00:10:17	00:13:08	00:56:55	00:13:07
FM	88%	91%	95.40%	94.70%	87%	87%	94.00%
500,000 Record (2 nodes)							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time		00:15:49	00:15:30	00:17:49	00:17:52	00:26:48	00:16:27
FM		79%	65%	84%	75%	75%	91%
750,000 Record (3 nodes)							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time		00:11:21	00:09:17	00:09:37	00:11:29	00:11:49	00:14:49
FM		75%	67.30%	64%	70%	60%	92%
1,000,000 Record (4 nodes)							
	Sc1	***Sc2***	***Sc3***	***Sc4***	***Sc5***	***Sc6***	***Sc7***
P. Time		00:12:19	00:10:27	00:11:11	00:12:38	00:10:30	00:15:35
FM		75%	59%	59%	70%	57%	91%

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

of the size 25,000 records, 50,000 records, 75,000 records, 100,000 records, 250,000 records respectively. From table 1 we can conclude that:

- Using the CA in scenarios (2, 3, 4, 5, 7) achieved better results than a single WA (scenario 1) in terms of efficiency and effectiveness. As fig. 4 illustrates, the usage of CA (scenarios 2, 3, 4, 5, 7) in the different scenarios achieved better performance time compared to the WA (first scenario).
- Scenario 6 had the worst performance time because no blocking techniques is implemented.
- Scenario 3 and 4 achieved the best performance time. Using Soundex and Stemming participate in reducing the performance time. As illustrated in fig.5, Scenario 3 and 4 have the best performance time.
- Scenario 5 and 6 achieved the best effectiveness as they have the heights F-measure. As illustrated in Fig.6, scenarios 5 and 6 have the highest F-measure using different dataset sizes.

B. Implementing the proposed seven scenarios using multi-nodes.

We used Apache Spark installed on multiple working nodes to implement the proposed seven scenarios. To test the proposed scenarios with the dataset of the size 500,000, 750,000, and 1000,000 records, we used 2, 3, and 4 working nodes in addition to the master node. From table 1 we can conclude that:

- Scenario 7 has the best F-measure since the size of the dataset grows, using the length filter results in minimizing the records of the dataset that will be compared as illustrated in fig. 7. In the length filter proposed in scenario 7, we only compare records within nearly the same length or may increase or decrease in length by a threshold of 20%. records of the dataset that will be compared. MinHashLSH is

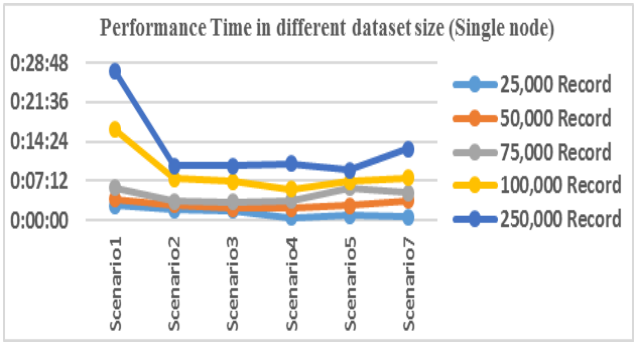


Fig. 4. The achieved performance time in the different seven scenarios

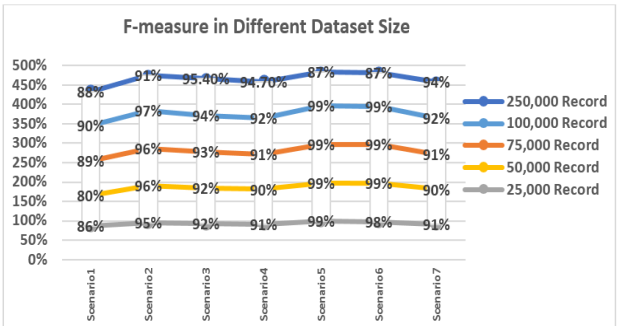


Fig. 5. The achieved F-measure in the different seven scenarios using 100,000 Records

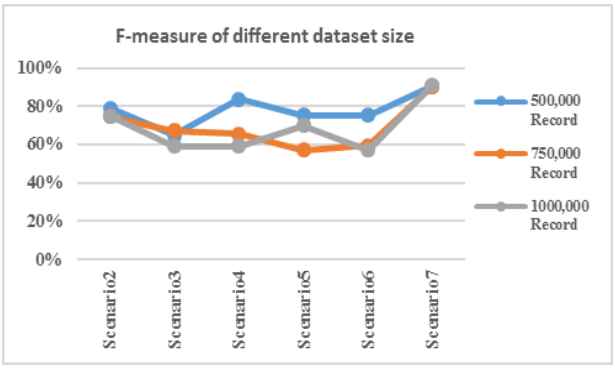


Fig. 6. The achieved F-measure in the different seven scenarios using multiple nodes.

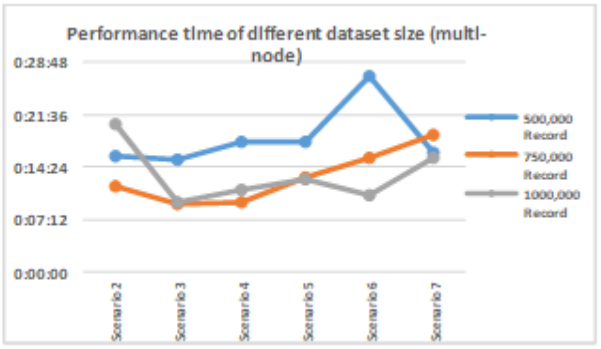


Fig. 7. The achieved performance time in the different seven scenarios using multiple nodes.

used to map the similar inputs to the same hash code with a higher probability than dissimilar ones. It takes num Hash Tables as a parameter. Increasing the number of hash tables will increase the accuracy of the model but will also increase the communication cost and the running time therefore, it is a compromise between computational resources and the accuracy of the approach. By using the length filter, we reduce the size of the dataset that will be compared and increase the accuracy of the Min HashL SH. Therefore, scenario 7 achieved the highest F-measure.

- As illustrated in Fig. 7, scenario 3 accomplished the best performance time in all the distributed working nodes (two, three, and four nodes). Therefore, adopting the Soundex technique in scenario 3 contribute to reducing the performance time.

V. CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel and efficient ER approach for big data implemented using Apache Spark. It consists of five subsequent phases. First, the dataset used in the ER is acquired. Second, data pre-processing is performed to standardize the data and remove all unwanted patterns. Third, the settings selection criteria are chosen. Fourth, the ER pipeline is constructed. Finally, the object is clustered such that similar ones are residing in the same cluster. The ER pipeline has five stages after accepting the setting selection criteria from the fourth phase. The pipeline could accept two types of attributes; the WA or the CA depending on the chosen scenario.

Retrieval Number: 100.1/ijitee.K950309101121
DOI: 10.35940/ijitee.K9503.09101121
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

It begins by generating the tokens composing the CA or the WA, then, producing n-grams from the generating tokens, using HashingTF to generate the vectors, using MinHashLSH to map similar inputs to the same hash code with a higher probability than dissimilar ones, and finally classifying the candidate objects using a similarity measure. To measure the similarity, we used Jaccard similarity distance. Blocking techniques have been used to only compare the records that are more likely to be matched. Seven different scenarios have been used and evaluated in the ER pipeline. In the proposed scenarios, we used different NLP techniques as Stemming and Soundex. Our proposed approach is a generic one that could accept different types of the dataset (structures, unstructured, and semi-structured).

It utilizes the relationship between attributes to predict the missing values of the attributes to enhance their quality. The proposed approach could be used with different languages as our dataset contains about 80 different languages and the approach could efficiently classify them. We used Hashing TF to generate the vectors. HashingTF is a transformer that takes a set of terms and converts them to fixed-length feature vectors. Hashing TF utilizes a feature hashing called hashing trick, which is a fast and space-efficient way of vectoring features. Using Soundex and Stemming before applying LSH help to reduce the length of features and thus feature vectors will be more space-efficient, which in turn enhances the performance time. To evaluate our proposed approach different dataset size has been used. To test the scalability, we used one, two, three, and four working nodes. Our evaluation shows that our proposed approach can distribute the similarity computation and classification among the computational resources and scale with the available working nodes. Experimental results show that using the CA achieves better results compared to the WA in terms of efficiency and effectiveness. The best performance time is achieved by scenario 3 and 4 because they adopt Soundex and Stemming techniques, which in turn participate in reducing the performance time of the approach. In case of multiple nodes, Scenario 7 achieved the best F-measure because as the size of the dataset grows, using the length filter results in minimizing the objects that will be compared and increase the accuracy of the MinHashLSH. For future work, we intend to evaluate our generic resolution algorithm against existing Entity Resolution algorithms using several publically available data sets. In addition, we want to extend the proposed ER approach with a ML module as we already created feature vectors that could be utilized with many predictions and clustering ML algorithms. Furthermore, we intend to apply the proposed approach in privacy-preserving record linkage (PPRL) applications.

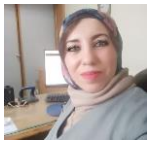
REFERENCES

1. Papadakis, G. Mandilaras, L. Gagliardelli, G. Simonini, E. Thanos, G. Giannakopoulos, S. Bergamaschi, T. Palpanas And M. Koubarakis, "Three-Dimensional Entity Resolution With Jedai," Elsevier, Vol. 93, 29 May 2020.
2. L. Kolb, H. Köpcke, A. Thor And E. Rahm, "Learning-Based Entity Resolution With Mapreduce," In Clouddb, 2011.
3. Gunawan, M. S. Lubis, D. Arisandi And B. Azzahry, "Data Pre-Processing In Record Linkage To Find The Same Companies From Companies From Different Databases," 2nd International Conference On Computing And Applied Informatics, Vol. 978, 2017.
4. R. M. Abd El-Ghafar, M. H. Gheith, A. H. El-Bastawissy And E. S. Nasr, "Record Linkage Approaches In Big Data: A State Of A State Of Art Study," In 13th International Computer Engineering Conference (Icenco), Cairo, 2017.
5. W. W. Cohen, P. Ravikumar And S. E. Fienberg, "A Comparison Of String Distance Metrics For Name-Matching Tasks," International Joint Conference On Artificial Intelligence, P. 73–78, 2003.
6. J. Mielke, "A Phonetically-Based Phonetic Similarity Metric," In Nels, 2009.
7. G. Mestre, C. E. S. Pires, D. C. Nascimento, A. R. Queiroz, V. B. Santos And T. B. Araujoa, "An Efficient Spark-Based Adaptive Windowing For Entity Matching," Journal Of Systems And Software, Vol. 128, P. 1:10, 3 March 2017.
8. Simonini, L. Gagliardelli, S. Zhu And S. Bergamaschi, "Enhancing Loosely Schema-Aware Entity Resolution With User Interaction," Hpcs, P. 860–864, July 2018.
9. Simonini, S. Bergamaschi And H. V. Jagadish, "Blast: A Loosely Schema-Aware Meta-Blocking Approach For Entity Resolution," In Pvldb, 2016.
10. R. Pita, C. Pinto, P. Melo, M. Silva, M. Barreto And D. Rasella, "A Spark-Based Workflow For Probabilistic Record Linkage Of Healthcare Data," In Edbt/Icdt Workshops, 2015.
11. M.-G. Chen And H.-J. Sui, "Parallel Entity Resolution With Apache Spark," In International Conference On Electronic, Control, Automation And Mechanical Engineering (Ecame 2017) , 2018.
12. Wang And S. Karimi, "Parallel Duplicate Detection In Adverse Drug Reaction Databases With Spark," In 19th International Conference On Extending Database Technology, Bordeaux, France, 2016.
13. B. Gatius And R. G. González, "Deduplication Of Universitat De Lleida Scholarly Data," 2017.
14. Alami, Y. Aassem And I. Hafidi, "Kf-Swoosh: An Efficient Spark-Based Entity Resolution Algorithm For Bigdata," Journal Of Physics: Conference Series, 2021.
15. Martinez, S. Cristobal And S. Belkoura, "Smart Data Fusion: Probabilistic Record Linkage Adapted To Merge Two Trajectories From Different Sources," In Eighth Sesar Innovation Days, 2018.
16. L. Gagliardelli, G. Simonini, D. Beneventano And S. Bergamaschi, "Sparkr: Scaling R Programs With Spark," In 22nd International Conference On Extending Database Technology (Edbt), Lisbon, Portugal, 2019.
17. L. Gagliardelli, S. Zhu, G. Simonini And S. Bergamaschi, "Bigdedup: A Big Data Integration Toolkit For Duplicate Detection In Industrial Scenarios," In Proc. Int. Conf. On Transdisciplinary Engineering (Te2018), 2018.
18. Papadakis, G. Koutrika, T. Palpanas And W. Nejdl, "Meta-Blocking: Taking Entity Resolution To The Next Level," Ieee Transactions On Knowledge And Data Engineering, Vol. 26, No. 8, P. 1964:1960, August 2014.
19. Papadakis, G. Mandilaras, L. Gagliardelli, G. Simonini, E. Thanos, G. Giannakopoulos, S. Bergamaschi, T. Palpanas And M. Koubarakis, "Three-Dimensional Entity Resolution With Jedai," Elsevier, 29 May 2020.
20. T. Sagi, A. Gal, O. Barkol, R. Bergman And A. Avram, "Multi-Source Uncertain Entity Resolution: Transforming Holocaust Victim Reports Into People," In International Conference On Management Of Data, 2016.
21. Karakasidis And G. Koloniari, "Private Entity Resolution For Big Data On Apache Spark Using Multiple Phonetic Codes," Big Data Recommender Systems, Vol. 1, 2019.
22. P. Brown And S. M. Randall, "Secure Record Linkage Of Large Health Data Sets: Evaluation Of A Hybrid Cloud Model," Jmir Medical Informatics, Vol. 8, No. 7, 2020.
23. Pita, C. Pinto, S. Sena And R. Fiaccone, "On The Accuracy And Scalability Of Probabilistic Data Linkage Over The Brazilian 114 Million Cohort," Ieee J Biomed Health Inform, Vol. 22, No. 2, P. 346–353, 2018.
24. X. Chen, R. Zoun, E. Schallehn, S. Mantha, K. Rapuru And G. Saake, "Exploring Spark-Sql-Based Entity Resolution Using The Persistence Capability," In International Conference: Beyond Databases, Architectures And Structures, Cham, 2018.
25. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh And A. H. Byers, "Big Data: The Next Frontier For Innovation, Competition, And Productivity.," 9 July 2012. [Online]. Available: Http://Www.Mckinsey.Com/Insights/Mgi/Research/Technology_And_Innovation/Big_Data_The_ Next_Frontier_For_Innovation.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication
© Copyright: All rights reserved.

26. P. Chen And C.-Y. Zhang, "Data-Intensive Applications, Challenges, Techniques And Technologies: A Survey On Big Data," Information Sciences, Vol. 275, P. 314:347, 2014.
27. Sorzano, J. Vargas And A. P. Montano, "A Survey Of Dimensionality Reduction Techniques," Arxiv:1403.2877, 2014.
28. M. Kuhn And K. Johnson, Applied Predictive Modeling., Springer, 2013, P. 488.
29. K.-T. Huang, Y. W. Lee And R. Y. Wang, Quality Information And Knowledge, New Jersey: Prentice Hall, 1999, Pp. 25-29.
30. P. Ballou And H. L. Pazer, "Modeling Data And Process Quality In Multiinput, Multi-Output Information Systems," Management Science, Vol. 31, No. 2, Pp. 150-162, 1985.
31. P. Ballou, H. L. Pazer, S. Belardo And B. Klein, "Implications Of Data Quality For Spreadsheet Analysis," Acm Sigmis Database: The Database For Advances In Information Systems, Vol. 18, No. 3, Pp. 13-19, March 1987.
32. T. Churches, P. Christen, K. Lim And J. X. Zhu, "Preparation Of Name And Address Data For Record Linkage Using Hidden Markov Models," Bmc Medical Informatics And Decision Making , 2002.
33. S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez And F. Herrera, "Big Data Preprocessing: Methods And Prospects," Big Data Analytics, November 2016.
34. D. Gunawan, M. S. Lubis, D. Arisandi And B. Azzahry, "Data Pre-Processing In Record Linkage To Find The Same Companies From Different Databases," Journal Of Physics: Conference Series, 2018.
35. S. Loeb , P. Morris, S. Dynarski , S. Reardon , D. Mcfarland And S. Reber, "Descriptive Analysis In Education: A Guide For Researchers," Ncee, Washington, 2017.
36. Papadakis, E. Ioannou, T. Palpanas, C. Niede And W. Nejdl, "A Blocking Framework For Entity Resolution In Highly Heterogeneous Information Spaces," Ieee Transactions On Knowledge And Data Engineering, Vol. 25, No. 12, Pp. 2665-2682, December 2013.
37. Kenig And A. Gal, "Mfiblocks: An Effective Blocking Algorithm For Entity Resolution," Information Systems, Vol. 38, No. 6, Pp. 908-926, 2013.
38. R. M. Abd El-Ghafar, A. H. El-Bastawissy, E. S. Nasr And M. H. Gheith, "An Efficient Multi-Phase Blocking Strategy For Entity Resolution In Big Data," International Journal Of Innovative Technology And Exploring Engineering (Ijitee), Vol. 9, No. 9, Pp. 254-263, July 2020.
39. Papadakis, D. Skoutas, E. Thanos And T. Palpanas, "A Servay Of Blocking And Filtering Techniques For Entity Resolution," Association For Computing Machinery (Acm), 2019.
40. Skoutas , T. Vergoulis, K. Patroumpas, S. Chatzopoulos, A. Zeakis , G. Chatzigeorgakidis, L. Shimomura , N. Yakovets, G. Fletcher, H. Shahrivari , O. Papapetrou And T. Spinner, "Similarity Search, Entity Resolution," Horizon 2020, 2020.
41. Wang, H. T. Shen, J. S. Song And J. Ji, "Hashing For Similarity Search: A Survey," Corr, 2014.
42. Y. W. Yu And W. M. Griffin, "Hyperminhash: Minhash In Loglog Space," Journal Of Latex Class Files , 2019.
43. M. Pham And T. L. X. Vu, "Elodu: Entity Resolution In Big Data," Worcester Polytechnic Institute, 2015.
44. S. P. Benny, S. V. And P. A. , "Hadoop Framework For Entity Resolution Within High Velocity Streams," In International Conference On Computational Modeling And Security (Cms), 2016.
45. J. Feigenbaum, "A Machine Learning Approach To Census Record Linking," 2016.
46. P. Dahiya And D. K. Srivastava, "Network Intrusion Detection In Big Dataset Using Spark," In International Conference On Computational Intelligence And Data Science, 2018.
47. Moir And J. Dean, "A Machine Learning Approach To Generic Entity Resolution In Support Of Cyber Situation Awareness," In Proceedings Of The 38th Australasian Computer Science Conference (Acsc 2015), 2015.
48. C.-J. Schild, S. Schultz And F. Wieser, "Linking Deutsche Bundesbank Company Data Using Machine-Learning-Based Classification," Deutsche Bundesbank Research Data And Service Centre, 2017.
49. Omran, A. A. Salman And A. Engelbrecht, "An Overview Of Clustering Methods," Intelligent Data Analysis, Vol. 11, No. 6, Pp. 583-605, 2007.
50. X. L. Dong And D. Srivastava, "Entity Resolution," Encyclopedia Of Database Systems, 2016.
51. Meyer, H. Naessens And B. D. Baets, "Algorithms For Computing The Min-Transitive Closure And Associated Partition Tree Of A Symmetric Fuzzy Relation," European Journal Of Operational Research, Vol. 155, No. 1, Pp. 226-238, 2004.
52. B. Li, Entity Resolution Over Graphs, Australia, 2014.
53. Nentwig, A. Groß, M. Moller And E. Rahm, "Distributed Holistic Clustering On Linked Data," In On The Move To Meaningful Internet Systems. Otm 2017 Conferences, 2017.
54. B. C And B. B. P. M, "Data Linkage For Big Data Using Hadoop Mapreduce," In Ijcst, 2015.
55. Kejriwal, "Entity Resolution In A Big Data Framework," In Twenty-Ninth Aaai Conference On Artificial Intelligence, Austin, Texas, 2015.
56. Efthymiou, K. Stefanidis And V. Christophides, "Minoan Er: Progressive Entity Resolution In The Web Of Data," In 19th International Conference On Extending Database Technology, Edbt, Bordeaux, France, 2016.
57. "Sparker: Scaling Entity Resolution In Spark," In 22nd International Conference On Extending Database Technology (Edbt), Lisbon, Portugal, 2019.
58. "Bigdedup: A Big Data Integration Toolkit For Duplicate Detection In Industrial Scenarios," In Proc. Int. Conf. On Transdisciplinary Engineering (Te2018), 2018.
59. Saeedi, M. Nentwig, E. Peukert And E. Rahm, "Scalable Matching And Clustering Of Entities With Famer," Complex Systems Informatics And Modeling Quarterly (Csimq), No. 16, P. 61–83, September/October 2018.
60. M. Ektefa, F. Sidi, H. Ibrahim, M. A. Jabar And S. Memar, "A Comparative Study In Classification Techniques For Unsupervised Record Linkage Model," Journal Of Computer Science, Vol. 7, No. 3, Pp. 341-347, 2011.
61. Jurek-Loughrey And D. P, Semi-Supervised And Unsupervised Approaches To Record Pairs Classification In Multi-Source Data Linkage, Switzerland: Springer, 2019.
62. M. Michalowski, S. Thakkar And C. A. Knoblock, "Exploiting Secondary Sources For Unsupervised Record Linkage," In Proceedings Of The 30th Vldb Conference, Canada,, 2004.
63. S. Sheth And A. R. Deshpande, "A Decision Tree Based Record Linkage For Recommendation Systems," International Journal Of Engineering Research And General Science ,, Vol. 3, No. 4, 2015.
64. K. Qian, L. Popa And P. Sen, "Active Learning For Large-Scale Entity Resolution," In Cikm, New York, 2017.

AUTHORS PROFILE



Randa Mohamed, is a master holder in computer science & information system. Currently, she is a PDH researcher in faculty of graduate studies for statistical research at Cairo University, Egypt. She is working as a Project manager at ministry of finance (IT Department). Her research interests include Big data, record linkage, natural language processing, machine learning, and data integration.

Ali H. El-Bastawissy, is recently the Dean of faculty of Computer Science in Modern Science & Art (MSA) University. He has written and published more than 50 articles and studies in Data Modeling and storage, Data Integration, Business Intelligence, and Corporate Information Strategies. He has over 35 years of experience designing and implementing business intelligence and enterprise integration solutions for dozens of Middle East Ministries and Governmental Associations to help them accelerate decision-making and improve corporate performance. Prof. El-Bastawissy supervised more than 35 master studies and about 10 PhD theses during his academic research life.

Eman S. Nasr, is a PHD degree holder in computer science. She is currently an independent Scholar. Her research interests include requirements engineering, embedded software systems, crowdsourcing, systems analysis, Web services, computer aided instruction, computer based training, decision making, feature extraction, formal specification, formal verification, genetic algorithms, human computer interaction, artificial intelligence, natural language processing, quality of service, software and engineering.

© Copyright: All rights reserved.

Mervat H. Gheith, is currently an associate professor in the department of computer science in the faculty of graduate studies for statistical research at Cairo University, Egypt. Her research interests include crowdsourcing, human-computer interaction, artificial intelligence, natural language processing.

© Copyright: All rights reserved.