A New Technique of Selecting an Optimal Blocking Method for Better Record Linkage

Kevin O'Hare[a], Anna Jurek[a], Cassio de Campos[a,b]

[a] *School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Computer Science Building, 18 Malone Road, BT9 5BN Belfast, United Kingdom*
[b] *Buys Ballotgebouw, Utrecht University, 3584 CC Utrecht, The Netherlands*

Abstract

Record linkage, referred to also as entity resolution, is the process of identifying pairs of records representing the same real world entity (e.g. a person) within a dataset or across multiple datasets. In order to reduce the number of record comparisons, record linkage frameworks initially perform a process referred to as blocking, which involves splitting records into a set of blocks using a partition (or blocking) scheme. This restricts comparisons among records that belong to the same block during the linkage process. Existing blocking methods are often evaluated using different metrics and independently of the choice of the subsequent linkage method, which makes the choice of an optimal approach very subjective. In this paper we demonstrate that existing evaluation metrics fail to provide strong evidence to support the selection of an optimal blocking method. We conduct an extensive evaluation of different blocking methods using multiple datasets and some commonly applied linkage techniques to show that evaluation of a blocking method must take into consideration the subsequent linkage phase. We propose a novel evaluation technique that takes into consideration multiple factors including the end-to-end running time of the combined blocking and linkage phases as well as the linkage technique used. We empirically demonstrate using multiple datasets that according to this novel evaluation technique some blocking methods can be fairly considered superior to others, while some should be deemed incomparable according to those factors. Finally, we propose a novel blocking method selection

Email addresses: `kohare08@qub.ac.uk` (Kevin O'Hare), `a.jurek@qub.ac.uk` (Anna Jurek), `c.decampos@uu.nl` (Cassio de Campos)

procedure that takes into consideration the linkage proficiency and end-to-end time of different blocking methods combined with a given linkage technique. We show that this technique is able to select the best or near best blocking method for unseen data.
Keywords: Record linkage, blocking, entity resolution

1. Introduction

Record Linkage (RL) is a process of identifying and linking pairs of records representing the same real world entity. An overview of a general RL process is demonstrated in Figure 1. As the number of record pairs that require comparison during linkage grows exponentially with dataset sizes, linkage often incurs great computational expense even for moderately sized datasets. For this reason, a blocking phase is implemented prior to linkage to reduce the otherwise high computational cost of exhaustively comparing all record pairs.

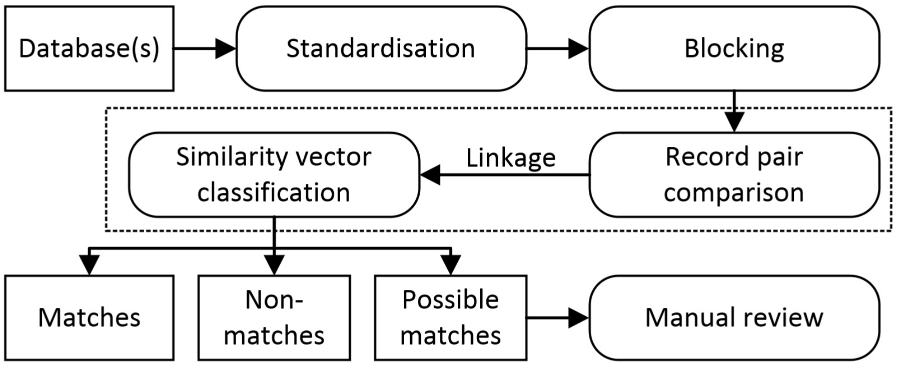


Figure 1: General overview of record linkage process.

Blocking is a process of dividing records into groups (blocks) in such a way that records within each group hold a high chance of being linked in the subsequent linkage process. Following the blocking process, linkage is performed exclusively upon the record pairs within each of the generated blocks.

During a blocking process a set of blocking keys is used to determine which records should be placed in the same block. Consider a dataset of records $R = r_1, ..., r_n$, where each record comprises values it takes for attributes from a scheme $A = a_1, ..., a_m$. Accordingly, we can represent a record r_i as $[r_{i1}, ..., r_{im}]$, where r_{ij} is the value that the i^{th} record takes for the j^{th} attribute. A blocking key is defined as follows.

Definition 1.1. *(Blocking key) A blocking key is an $\langle a_j, h\rangle$ combination where $a_j \in A$ is an attribute and h is an indexing function. For each $r_i \in R$, h takes r_{ij} as an input and provides a set of values, referred to as blocking key values (BKVs), as an output.*

For example, the blocking key $\langle \textit{Name}, \textit{Contain common tokens}\rangle$ applied to a record containing "Information Systems Journal" in the $Name$ attribute field would generate a BKV set containing three BKVs {"Information", "Systems", "Journal"}. BKVs determine into which block(s) records are placed, with each unique BKV referring to a specific block. Our example record would therefore be placed in three different blocks, each associated with one of the three aforementioned BKVs.

A good blocking method places many matching record pairs and few non-matching record pairs into the generated blocks thus allowing for an efficient subsequent linkage phase. A number of different linkage methods exist which classify each record pair within each block as either match or non-match based on the similarity between them [12, 15, 16, 21, 24]. Due to the complexity of datasets (i.e. missing values, typographical errors, acronyms, initialisations, etc.) a single blocking key is rarely likely to capture all matching record pairs efficiently, therefore multiple blocking keys may be needed in the form of a blocking scheme.

Definition 1.2. *(Blocking Schemes) Given a set of individual blocking keys, $K = k_1, ..., k_{k'}$, a blocking scheme is a combination of blocking keys, which can be disjunctive i.e. $\langle k_i\rangle \cup \ldots \cup \langle k_j\rangle$, conjunctive i.e. $\langle k_i\rangle \cap \ldots \cap \langle k_j\rangle$ or of disjunctive normal form i.e.$\langle\langle k_i\rangle \cap \ldots \cap \langle k_j\rangle\rangle \cup \ldots \cup \langle\langle k_{i'}\rangle \cap \ldots \cap \langle k_{j'}\rangle\rangle$*

Blocking schemes may be created manually [13, 15] or automatically learned [2, 20, 27] using a blocking scheme learning algorithm and labelled data.

Blocking methods are commonly evaluated with labelled data (with known matching status of each record pair) using evaluation metrics such as *reduction ratio (RR)*, *pairs completeness (PC)* and/or a harmonic mean $F_{\text{RR,PC}}$ of RR and PC [18].

Definition 1.3. *(Reduction Ratio) For two datasets, A and B, reduction ratio is defined as:*

$$RR = 1 - \frac{N}{|A| \times |B|}, \tag{1}$$

where $|A|$ and $|B|$ are the sizes of respective datasets and $N \leq (|A| \times |B|)$ is the number of record pairs formed by a blocking method.

RR indicates how much the comparison space is reduced after the blocking phase. For example, if a potential comparison space of 1,000,000 record pairs was reduced by blocking to 5,000 record pairs, that would equate to $RR = 1 - (5,000/1,000,000) = 0.995$.

Definition 1.4. *(Pairs Completeness) Pairs completeness is defined as:*

$$PC = \frac{N_m}{|M|}, \tag{2}$$

with $N_m \leq |M|$ being the number of matching record pairs contained within the reduced comparison space after blocking and $|M|$ being the number of matches within the entire dataset.

PC is the ratio of matching record pairs found within the formed blocks. One can notice that there is a trade-off between RR and PC. Comparing all record pairs (placing all the records in the same block) minimises RR but maximises PC, whereas performing no comparisons at all (placing each record in an individual block) maximises RR and minimises PC. Ideally one looks for a blocking scheme that maximises both RR and PC. A commonly applied evaluation metric, which balances the trade-off between RR and PC, is the harmonic mean of RR and PC.

Definition 1.5. *(Harmonic mean of RR and PC) For a given RR and PC, the harmonic mean is defined as:*

$$F_{RR,PC} = \frac{2 * RR * PC}{RR + PC}. \tag{3}$$

In this paper we make the following contributions: (1) We compare existing blocking scheme learning methods using results from the respective papers to show that current blocking evaluation metrics are insufficient when performed independently of a subsequent linkage phase. Analysis of these results show that no blocking method is superior to the others in every instance, and that the choice highly depends on which evaluation metric is prioritised. (2) We propose a novel technique that evaluates blocking methods as part of an RL framework (i.e. takes the quality and runtime of the subsequent linkage into consideration) and visualises the results graphically. This allows an optimal blocking method to be easily identified according to multiple factors, including resources (in particular running time), datasets and linkage method. (3) We propose a new selection technique that uses results obtain by blocking methods on known labelled datasets to select an optimal blocking method for a new unlabelled dataset for a given RL method. We perform a number of experiments using different blocking methods and some of the commonly used RL methods. We compare the results of our selected methods against all others on multiple datasets to show that an optimal or near optimal blocking method is selected in every case.

2. Relevant Work

Automatic blocking scheme learning approaches [2, 20, 27] commonly evaluate an initial set of individual blocking keys against a set of labelled data. The best individual keys, according to a predetermined criterion, continue to iteratively form blocking schemes with remaining individual keys often re-ranked between iterations. These schemes are then evaluated against labelled data using evaluation metrics. A blocking key or a blocking scheme is commonly evaluated with reduction ratio (RR), pairs completeness (PC) and harmonic mean of RR and PC ($F_{\mathrm{RR,PC}}$), following the blocking phase.

The supervised approach in [27] ranks individual keys with PC above a predetermined threshold by RR. Each top key is extended by other keys as conjunctions so RR improves while maintaining PC above the threshold. This continues until RR no longer improves for each conjunction. The idea is that although each individual conjunction may only cover a certain proportion of the matches, their disjunction will collectively detect most if not all matches. The proficiency of learned blocking schemes against different datasets are presented using RR and PC. While this paper presents good results, they are presented independently of computational run-time.

Another supervised approach [2] ranks keys by their ratio of detected matches to non-matches. Top keys are then iteratively applied to the labelled record pairs as a disjunctive blocking scheme until a predetermined proportion of labelled positives are detected. Disjunctive normal form schemes may also be learned by iteratively extending each top key by others so that the ratio is maximally improved. This continues until a conjunction of desired length is generated. The individual keys are supplemented by the conjunctions formed at each iteration. The supplemented set is then ranked and iteratively applied to the labelled record pairs as a disjunction of conjunctions blocking scheme until a predetermined proportion of labelled positives are detected.

Learned schemes are applied to datasets with results presented using RR versus PC curve graphs. This work mentions run-time by presenting the average blocking time per approach against both datasets that were evaluated. The average blocking time is measured at maximum achieved PC and defined as the total time taken to construct the blocks and to generate the candidate pairs. The paper does not take into consideration the time taken for a subsequent linkage phase.

With the previous two approaches, labelled training data is assumed to be available, which usually requires considerable time and effort to produce. The blocking scheme learning approach of [20] improves upon this by generating its own labels from the target dataset. To achieve this they first group records according to shared common tokens, for example containing “Avenue” within their address attribute value. Within each grouping, a window of predetermined size is slid over the contained records. Record pairs within the window at any given time are then compared using the (log) Term Frequency-Inverse Document Frequency (TF-IDF) measure (we discuss it further in Eq.(8)). The top d pairs are labelled as positives and the bottom nd as negatives, where d and nd are the two parameters of the method. Blocking keys are then individually evaluated by their Fisher Score [14] of agreements/disagreements on the labelled data. In addition to RR and PC, $F_{\text{RR,PC}}$ is also used to evaluate the performance of learned schemes against three datasets. Running time is mentioned but only in terms of the complexity for the labelled data generation phase.

The work presented in [2, 20, 27] detail automatic approaches for learning of standard blocking schemes for homogeneously structured datasets. Although other blocking scheme learning approaches exist, they either necessitate a domain expert, incorporate non-standard blocking approaches, are specific to non-homogeneously structured datasets, or focus on other aspects of the RL process such as improving the efficiency of the linkage phase. One such alternative approach [19] to standard automatic blocking scheme learning algorithms maps entities to a multi-dimensional index dependent upon the similarity metrics and blocking functions that are used. Nearby entities in this multi-dimensional space are then compared to one another. In both evaluations the proficiency of the proposed approach surpasses that of standard blocking with perfect PC being achieved in less comparisons. However, the use of multi-dimensional indices comes at a greater overhead cost than standard blocking, making the approach only suitable for datasets of much lower dimension than is commonly seen in real-world cases.

A recent clustering based approach [17] uses blocking keys to form blocks with regulated size. Between each iteration a block quality and block size trade-off is used to determine which overly small blocks should be merged and which overly large blocks should be further partitioned. The authors state that although their blocking scheme learning approach takes longer than that of the baselines, they achieve superior RR because of the regulated blocks sizes. They also detail that in their evaluations they manually determined the blocking keys and their order by which to further partition overly large blocks. Automatic learning of blocking keys is left for future work.

A limitation of current RL evaluation techniques is identified in [5]. Since different approaches are often evaluated using a variety of different metrics, a general comparison and an overall conclusion are often impossible. They therefore advocate the use of precision-recall curve graphs as the most informative RL evaluation methodology. *Precision* (*Prec*), when referring to linkage, is the proportion of correctly classified matching record pairs.

$$Precision = \frac{|TP|}{|TP| + |FP|}. \tag{4}$$

Recall (*Rec*), when referring to linkage, is the proportion of correctly classified matches out of all known matches in the dataset.

$$Recall = \frac{|TP|}{|TP| + |FN|}. \tag{5}$$

When referring to the linkage phase $|TP|$ is the number of true positives (correctly classified matches), $|TN|$ is the number of true negatives (correctly classified non-matches), $|FP|$ is the number of false positives (non-matches classified as matches) and $|FN|$ is the number of false negatives (matches classified as non-matches). Precision-recall curve graphs have been used in a number of papers [3, 6, 21, 26], but they are only an indicator of RL proficiency post completion and do not take into account the amount of resources (i.e. time) to achieve such results.

In [24] different RL frameworks incorporating manually defined blocking schemes are compared using a variety of evaluation metrics. RR, PC, $F_{\text{RR,PC}}$, Recall, Precision and $F_{\text{Prec,Rec}}$ as well as time, Accuracy and Pairs Quality (referred to as PQ) are used. Accuracy is the total proportion of record pairs that are either correctly blocked (in the case of matches) or correctly not blocked (in the case of non-matches) out of all record pairs. PQ is the proportion of correctly blocked record pairs out of all blocked record pairs and is used as a surrogate for blocking precision.

$$PQ = \frac{\text{Total number of blocked matching record pairs}}{\text{Total number of blocked record pairs}}. \tag{6}$$

Time resources are indicated for each method's execution. However, these time values are used independently of the other metrics, making difficult any comparisons of proficiency over a fixed time period.

A comprehensive empirical survey of 17 blocking methods upon 6 popular real datasets and 7 synthetic datasets is carried out in [31]. The authors examine the robustness of the internal configurations and relative balance between effectiveness and time efficiency for each method upon each dataset. The effectiveness of the blocking methods are estimated using RR, PC and PQ. In their evaluations the subsequent linkage process is also taken into consideration but only in terms of the computational runtime required to perform all pairwise comparisons. For larger datasets this is estimated using the average time required to perform 10^8 comparisons.

A developing area of relevance is that of Meta-Blocking [28, 29, 30, 32]. Meta-blocking aims to improve RL efficiency by only performing linkage upon the blocks and/or record pairs within the blocks indicated as most likely to refer to matching record pairs. In [28] the authors observe that a record paired with many other records is unlikely to match with any during linkage. By constructing an entity index, which monitors which records belong to which blocks and vice versa, they exploit this observation in order to prioritise blocks and/or record pairs for linkage. Rather than perform all record pair comparisons of a block collection arbitrarily, as is done typically, linkage is only performed until a cost/gain approximation indicates that finding further matches is too costly.

More recently schema-agnostic blocking approaches have been improved using meta-blocking. Unlike standard blocking, schema-agnostic blocking approaches do not require any prior knowledge of a datasets schema (i.e. attribute columns) in order to be implemented. $Token$-Blocking is one such example in which records are blocked according to shared common token values regardless of where in each record the token is present. As such $token$-blocking has high PC but low precision as common tokens may be present in different attributes between records of different data sources. In [32] attributes between different data sources are clustered so that only record pairs with a common token between ”similar” attributes are considered. In their evaluations only the blocking and meta-blocking stages are evaluated with PQ used as a surrogate for precision alongside PC and $F_{\text{PQ,PC}}$. PQ is greatly improved whilst maintaining high PC with their approach performing better than other meta-blocking approaches and equally well as standard blocking approaches.

An alternative linkage efficiency improvement [33] merges and distributes any record pair classified as matching to all other blocks containing either record. This improves PC due to the transitive relation of matching record pairs. It also improves efficiency in a progressive convergent manner as the repeated merging, distribution and replacement of individual records with merged records in earlier blocks saves the overall processing time of all other blocks.

3. Problem Formulation

An issue with existing comparisons of blocking methods for RL is that different metrics have been used for the evaluation of different approaches (RR - Reduction Ratio, PC - Pairs Completeness, $F_{\text{RR,PC}}$ - Harmonic mean of RR and PC), rendering most conclusions subjective to a great degree [5]. A further issue is that a blocking method that scores highly by one metric may not be suitable for all users. For example, a method with especially high PC may also have poor RR. Such a method may be unusable by a user under time constraints as the linkage phase may take longer than they can afford (RR correlates with the time needed for the linkage). On the other hand, a blocking method with high RR may produce blocks that are poor in terms of PC. Therefore, many researchers adopt $F_{\text{RR,PC}}$, which acts as an indicator of balance between RR and PC. However, choosing a blocking method indicated as optimal by this metric may still result in an execution run-time longer than some users can afford, and arguably an approach which is allowed to run for a longer time should be expected to yield better results, since it has more computational resources at its disposal. To better demonstrate the aforementioned issue, Table 1 shows results collected from [2, 20, 27]. A and $A^{'}$ refer to the method proposed in [2] with a disjunctive and disjunctive normal form blocking scheme learned respectively. Likewise, B and $B^{'}$ stand for the method introduced in [20] in the two cases. For [27] only disjunctive blocking schemes were learned, C and $C^{'}$ in this case represent results for when 10% and 50% of labeled training data are used respectively. $F_{\text{RR,PC}}$ values were not given for A, $A^{'}$, C or $C^{'}$ in their respective works but are present in Table 1 as they were calculated from the RR and PC values. We present interval values for A and $A^{'}$, as opposed to precise values, since RR and PC were estimated from graphs. Although other datasets were used, we focus on results for datasets *Restaurant*, *Cora* and *Census*, since they are common to at least two of the aforementioned papers.

From the results that we have compiled in Table 1, we see that often different blocking methods are selected if one employs a particular metric to analyse them. For example, if prioritising by RR, then $B^{'}$ is selected for Restaurant, $A^{'}$ for Cora and C for Census. $F_{\text{RR,PC}}$ is seen as an indicator of how good a balance between RR and PC is

Algorithm from:		[2]	[2]	[20]	[20]	[27]	[27]
Dataset	Metric	A	$A^{'}$	B	$B^{'}$	C	$C^{'}$
Restaurant	PC	**1.0000**	**1.0000**	0.9554	0.9554	0.9348	0.9816
	RR	0.9990	0.9990	0.9800	**0.9993**	0.9957	0.9926
	$F_{\text{RR,PC}}$	**0.9995**	**0.9995**	0.9675	0.9768	0.9643	0.9871
Cora	PC	$[0.995, 0.997]$	**[0.997,0.998]**	0.9443	0.9443	–	–
	RR	$[0.94, 0.96]$	**[0.96,0.97]**	0.9330	0.9330	–	–
	$F_{\text{RR,PC}}$	$[0.965, 0.978]$	**[0.978,0.984]**	0.9386	0.9386	–	–
Census	PC	**1.0000**	**1.0000**	**1.0000**	**1.0000**	0.9913	0.9985
	RR	0.9916	0.9916	0.9916	0.9916	**0.9950**	0.9812
	$F_{\text{RR,PC}}$	**0.9958**	**0.9958**	**0.9958**	**0.9958**	0.9931	0.9898

Table 1: Reduction Ratio (RR), Pairs completeness (PC) and harmonic mean of RR and PC ($F_{RR,PC}$) over different datasets for a collection of approaches from the literature, as indicated in the first and second rows. A and $A^{'}$ respectively refer to the disjunctive and disjunctive normal form variations of the blocking scheme learning approach of [2]. This is also the case for B and $B^{'}$ and the approach of [20]. In [27] only disjunctive blocking schemes were learned, C and $C^{'}$ in this case represent results for when 10% and 50% of labeled training data are used respectively. Columns A and $A^{'}$ were obtained from the experiments presented in [20], except for the intervals of the centre three rows, which are from [2] itself.

achieved. In [20], such a metric is preferred over RR and PC for blocking method evaluation. An issue here however is that two blocking methods with the same measure of $F_{\text{RR,PC}}$ may have very different values for RR and PC. For example, for $F_{\text{RR,PC}}$=0.788 we may have RR=0.700 and PC=0.900, or RR=0.900 and PC=0.700. These two cases would result in very different linkage results for a large dataset, one being quicker but potentially detecting less matches and the other slower but potentially detecting a higher number of matches. This becomes especially evident in significantly large datasets in which a difference in RR of 0.001 could equate to millions of record pair comparisons.

Some related work presents the computational run-time of the subsequent linkage phase in addition to the blocking evaluation metrics [23, 31]. In the latter the computational runtime of the linkage phase is estimated for the larger datasets by using the average runtime for 10^8 record pairs. This gives a more informed comparison between blocking methods but fails to address another issue. Even with identical RR, PC and consequently $F_{\text{RR,PC}}$ values, the quality of the end result (i.e. post-linkage) relies on which record pairs were formed by the respective blocking method. One of the aims of this paper is to show that although $F_{\text{RR,PC}}$ of blocking indicates the proficiency of RL, it does not guarantee better linkage results than if another blocking method with lesser $F_{\text{RR,PC}}$ was applied.

In summary, there are multiple blocking evaluation problems that need to be addressed. Firstly, no single existing evaluation metric can effectively indicate the optimal blocking method for users with differing preferences (i.e. those with/without time constraints). Secondly, no existing evaluation metric can effectively indicate the overall quality of a blocking method considering multiple factors such as accuracy and time resources. Thirdly, no evaluation metric can guarantee one blocking method will perform better than another post-linkage. To properly evaluate blocking methods one must therefore take into account the quality of the post-linkage results, not just the blocking evaluation metrics and the computational runtime of the subsequent linkage.

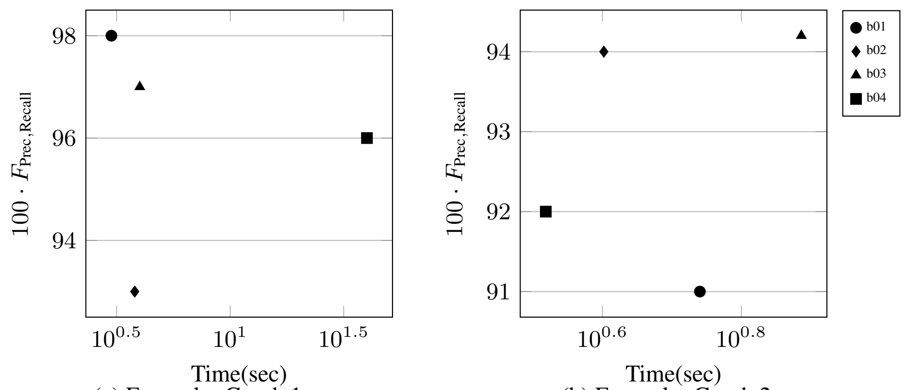
4. Proposed Approach

4.1. A New Technique for Evaluating Blocking Methods

We propose that the evaluation of blocking methods must take into consideration the proficiency of a subsequent linkage phase, as the performance of the latter is intrinsically related to the former. This is not only because of the running time but also specific blocking methods may be more suited to specific linkage approaches. We therefore propose to perform the evaluation of blocking methods as part of RL frameworks. For this purpose we apply $F_{\text{Prec,Rec}}$ (F-measure calculated post-linkage), which is the harmonic mean of Precision and Recall (Eq.(7)), in conjunction with the overall running time (i.e. blocking and linkage) to evaluate the performance of different blocking methods applied with a single linkage method. This contrasts with other work [2, 20, 27] where blocking is allowed to run until completion and blocking quality results only are presented, often as the optimal or average across multiple parameters, making it harder to make objective conclusions.

$$F_{\text{Prec,Rec}} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{7}$$

In order to better indicate how well each blocking method performs with respect to each other we plot the two metrics on a graph as shown in the exemplar Figures 2a and 2b.



(a) Exemplar Graph 1 (b) Exemplar Graph 2

Figure 2: A toy example of $F_{\text{Prec,Rec}}$ vs Time(sec) graph for two datasets. b_{01},...,b_{04} represent different blocking methods, $F_{\text{Prec,Rec}}$ are the subsequent post-linkage proficiency results (Eq.(7)) of each blocking method for each dataset (given a particular linkage technique) and Time(sec) are the end-to-end time values of the combined blocking and linkage phases.

In each graph a point is plotted to represent the $F_{\text{Prec,Rec}}$ and Time(sec) results of different combinations of blocking methods and linkage techniques upon a dataset. In our exemplar figures points for 4 different blocking methods were deliberately placed using fictitious results in order to demonstrate two typical cases that may occur. Looking at these graphs one can identify which blocking methods outperform others. We define an outperforming blocking method as follows.

Definition 4.1. *(Pareto-outperforming blocking method) Blocking method b_i Pareto-outperforms blocking method b_j for a given dataset and a given linkage method if b_i achieves higher $F_{Prec,Rec}$ value in less time than b_j.*

In Figure 2a, method b_{01} can be clearly seen to Pareto-outperform all other methods making it the obvious best blocking method. In some cases, however, we are not able to make an unambiguous conclusion as to which blocking method is best as some might have better $F_{\text{Prec,Rec}}$ than others but poorer overall running time. In Figure 2b for example, a user may prefer either b_{02} or b_{04} as they Pareto-outperform b_{01}, or else b_{03} as it has the highest $F_{Prec,Rec}$ but also takes the longest. In such cases, it should be a user's decision about the optimal approach to use according to availability of time and possible required accuracy i.e. choose the method with highest proficiency within the time allowed. For our purposes we visually consider an optimal blocking method for a dataset to be the blocking method in the top-most left corner of a respective $F_{\text{Prec,Rec}}$ vs Time graph. This is because this blocking method achieves the best balance of maximising $F_{Prec,Rec}$ and minimising Time in comparison to all other methods. Using this idea, in Figure 2b we would therefore consider b_{02} to be an optimal blocking method despite there existing other faster or more proficient blocking methods.

Definition 4.2. *(Optimal blocking method) For a set of blocking methods* $b_1...b_n$ *and a* $F_{Prec,Rec}$ *vs Time graph, the optimal blocking method is defined as:*

$$\arg\min_{b_1,\ldots,b_n} \sqrt{(0 - b_i(Time))^2 + (1 - b_i(F_{\mathit{Prec,Rec}}))^2}$$

4.2. A Technique for Selecting Optimal Blocking Methods for a New Dataset

In the previous section we detail a technique that takes the post-linkage results and end-to-end time into consideration when comparing the performance of different blocking methods for a dataset. This is only possible when a sufficient number of labelled records from the dataset is available, which is inherently not the case for real world users. We propose a blocking method selection technique that uses other labelled datasets to select an optimal (or near-optimal) blocking method (for a given linkage technique) for a new and unlabelled dataset. With the proposed approach we model the relation among different blocking methods as a graph, which we refer to from now on as a dominance graph.

Definition 4.3. *(Dominance Graph) Given a set of different blocking methods* $B = b_1, ..., b_m$ *and a collection of different datasets* $D = d_1, ..., d_n$*, the dominance graph is* *a graph with m nodes where for each dataset $d \in D$ there exists a directed edge from node b_i to b_j if b_j Pareto-outperforms b_i on d.*

With a dominance graph we can visualise the relations between different blocking methods across multiple datasets. A very simple example of such a graph is presented in Figure 3 where b_1 to b_4 are nodes representing different blocking methods and the directed edges between them represent which methods Pareto-outperform others in at least one of the multiple labelled datasets.

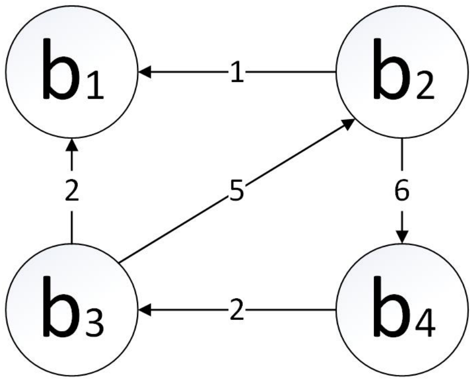


Figure 3: Example of a simple dominance graph where $b_1, ..., b_4$ are nodes representing different blocking methods. Numbered edges between nodes indicate in how many of the evaluated datasets one node Pareto-outperforms the other. Edges are directed towards the superior node.

Edges are directed from the outperformed node to the outperforming node and may overlap in a dominance graph if the same relation occurs in multiple datasets. In Figure 3 a number overlays each edge to indicate in how many of the datasets this relation occurs. At the same time, there may be pairs of nodes that are not connected by any edge. We further define a dominating node.

Definition 4.4. *(Dominating Node) For a given dominance graph we say that method (node) b_i dominates method (node) b_j if there is a directed path from b_j to b_i and there is no directed path from b_i to b_j.*

We define a non-dominated node as follows.

Definition 4.5. *(Non-Dominated node) We say that a method (node) b_i is non-dominated in a dominance graph if there does not exist a node in the graph that dominates b_i.*

In Figure 3 nodes b_2, b_3 and b_4 are dominated and b_1 is non-dominated. The existence of a non-dominated node is guaranteed within a dominance graph. A strongly connected component (SCC) is a subset of nodes where each node forms a path with every other node within the subset. In Figure 3 the subsets (b_1) and (b_2, b_3, b_4) each form an SCC. A graph of SCCs forms a directed acyclic graph. Directed acyclic graph theory states that there must exist at least one SCC that has no children SCCs (referred to as leaves of the directed acyclic graph). In Figure 3 the SCC (b_1) would therefore be considered the only leaf of the directed acyclic graph. Looking at the leaf SCC, any original node that has no children SCCs must therefore be non-dominated as the only incoming edges to that node come from the other nodes within the SCC. In our case b_1 is the guaranteed non-dominated node.

Our intuition is that non-dominated nodes from a dominance graph generated with multiple datasets are likely to be good for other datasets as well. As there may be multiple non-dominated nodes in a dominance graph an additional selection step is required. This consists of selecting the non-dominated node with the largest $Incoming - Outgoing$ number of edges value. In Figure 3 method b_1 would therefore be selected.

4.2.1. Blocking Method Selection Process

The pseudo-code of the blocking method selection process is outlined in Algorithm 1. Given n labelled datasets, m blocking methods and a particular linkage technique an optimal blocking method for a given linkage method can be selected using the selection technique as follows.

i. (Lines 2-4) Apply each blocking method to the n labelled datasets and perform their respective subsequent linkage phases using the linkage technique. Record the end-to-end time values of the combined blocking and linkage phases as well as the post-linkage $F_{Prec,Rec}$ values.

ii. Using these results form a $F_{Prec,Rec}$ vs Time graph for each labelled dataset.

Algorithm 1: Blocking Method Selection

```
Input: Set of labelled datasets, $\mathbf{D} = d_1, ..., d_n$
       Set of blocking methods, $\mathbf{B} = b_1, ..., b_m$
       Linkage technique, $Link$
Output: Optimal blocking method, $b_{Optimal}$
1  $Results$ = ∅
2  foreach $d_i \in \boldsymbol{D}$ do
3      foreach $b_j \in \boldsymbol{B}$ do
4          $Results_{i,j} = Link(b_i(d_j))$
5          $Results = Results \cup Results_{i,j}$
6  foreach $b_j \in \boldsymbol{B}$ do
7      Add node $b_j$ to the dominance graph
8      foreach $d_i \in \boldsymbol{D}$ do
9          if $\exists b_u \in B$ such that $Results_{i,u}$ outperforms $Results_{i,j}$ then
10             Add edge $b_u \leftarrow b_i$ to the dominance graph
11 ND = ∅
12 foreach $b_i \in \boldsymbol{B}$ do
13     if $b_i$ is non-dominated then
14         ND = ND $\cup\ b_i$
15 $b_{Optimal}$ =$\arg\max_{b \in \mathbf{B}} b_{Incoming-Outgoing}$
16 Return $b_{Optimal}$
```

iii. If a singular blocking method can be clearly seen to outperform (according to Definition 4.2) every other blocking method in every graph, select that blocking method to use with the linkage technique upon the unlabelled dataset. Otherwise perform steps *v* to *iix*.

iv. (Lines 6-10) Using the $F_{Prec,Rec}$ vs Time graphs from step iii, form a dominance graph as per Definition 4.3. For every connected pair of nodes (blocking methods) within the dominance graph, indicate with an overlaying number in how many of the datasets one method Pareto-outperforms the other.

v. (Lines 11-14) Select all non-dominated nodes according to Definition 4.5.

vi. (Line 15) If only one Non-Dominated node remains, select the respective blocking method of that node. If multiple Non-Dominated nodes remain select the respective blocking method of the node with the largest $Incoming - Outgoing$ number of directed edges.

vii. Use the respective blocking method of the selected node along with the linkage technique upon the unlabelled dataset.

5. Experimental Evaluation

All algorithms were coded using Java Eclipse Mars.1. Evaluations were ran using a Dell Optiplex 9020 with 16G of RAM, an Intel(R) Core(TM) i7-4790 with 3.60GHz and 64x Windows 7 Enterprise.

Blocking Methods

In our experiments we used the blocking algorithms which were described in detail in section 3. With each of the three algorithms an initial set of blocking keys is applied to a subset of labelled data. Blocking keys which pass a satisfaction criteria are then combined to form longer conjunctions. This continues iteratively until conjunctions of a maximum predetermined length are constructed. Keys and their conjunctions are then ranked according to a criteria unique to each paper. The ranked blocking keys and their conjunctions are then applied iteratively as a blocking scheme to the labelled data until a predetermined proportion of labelled positives are detected. Each of the algorithms was implemented to our best understanding of the papers. [2] limited their generated blocking methods to use conjunctions with a maximum length of up to 4 keys (see definition 1.2). Longer conjunctions potentially lead to more efficient blocks but come at a cost of greater runtime for their learning and evaluation. In our experiments, we apply four different maximum conjunction lengths (i.e. 1 to 4 keys) for each algorithm. Consequently, in total 12 variants of the blocking methods were evaluated for each dataset (3 blocking scheme learning algorithms $\times$ 4 possible conjunction lengths). In the case of [27] only disjunctive normal form blocking methods were previously generated and evaluated. Therefore by having a variant in which there is a maximum conjunction length of 1 we have now added the ability for disjunctive blocking methods to also be generated by this particular approach.

Linkage Techniques

Three different linkage techniques have been combined with the blocking methods to form distinct RL frameworks. First was a hypothetical instantaneous perfect linkage technique where each record pair is perfectly classified and no time is incurred for comparison. The overall time values by this linkage technique therefore only represent the blocking process as well as any pre-processing (e.g. standardisation). As classification is perfect the $F_{\text{Prec,Rec}}$ (harmonic mean of precision and recall) values for this linkage technique are therefore given using the recall value obtained by the blocking method and precision equal to one. We use this hypothetical linkage technique to better demonstrate the disparity of linkage quality and end-to-end time before and after application of an actual linkage technique. As the second linkage technique we employ LibSVM [8], a popular Support Vector Machine (SVM). SVMs are a supervised classification model learning technique that form an optimal separating hyper-plane with maximal margins between the positive and negative training comparison vectors. SVMs have been used in many RL related works [1, 9, 10, 22, 32]. A particularly popular example is that of the MARLIN (Multiply Adaptive Record Linkage with INduction) system [4] which is often used as a baseline in RL related papers [7, 21, 25, 27]. For the third linkage technique we adopt a rule based approach [16] which uses Log TF-IDF (Term Frequency Inverse Document Frequency) for measuring similarity between records. The Log TF-IDF measure [20] is formally defined as

$$\text{sim}(t_1, t_2) = \sum_{q \in t_1 \cap t_2} w(t_1, q) \cdot w(t_1, q), \tag{8}$$

where

$$w(t, q) = \frac{w'(t, q)}{\sqrt{\sum_{q \in t} w'(t, q)^2}}, \tag{9}$$

and

$$w'(t, q) = \log(tf_{t,q} + 1) \cdot \log(\frac{|R|}{df_q} + 1) \tag{10}$$

where (t_1, t_2) represents a record pair, $w(t, q)$ is the normalised TF-IDF weight of a term q in a record t, $tf_{t,q}$ represents the term frequency of q in t, $|R|$ is the total number of records in the dataset R, df_q is the document frequency of the term q in the cohort, that is, how many records in the dataset contain t. In order to classify record pairs by this linkage technique a similarity threshold value is required. For each dataset an optimal TF-IDF linkage threshold value was set.

Validation

We implement ten-fold cross-validation to evaluate each dataset by each framework. The results presented in the respective tables are thus the accumulated average findings across the respective ten folds. If for a fold, a blocking method was unable to learn a blocking scheme under some given parameters, its result is deemed as zero and still contributes to the average.

Indexing functions

Different sets of indexing functions (applied for constructing different blocking keys) were used in [27] for each dataset when compared to [2] and [20]. For the purpose of a fairer comparison, we use the same 25 indexing functions commonly used between [2] and [20].

Datasets

The evaluation was performed using 2 of the datasets common to [2] and [20] (*Restaurant* and *Cora*) as well as additional larger datasets. The datasets used have varying characteristics represented by their differing column values in Table 2. The column Dedup/RL details whether a dataset is used for deduplication (within a single dataset) or record linkage (across 2 datasets). Synth/Real signifies whether a dataset is a real publicly available labelled dataset commonly used in RL or a synthetic dataset that we generated for our evaluations. To generate synthetic data we use a modified version of the synthetic data generator from [11]. Clean/Dirty respectively signify whether a record may only match with at most one other record or may match with multiple other records. Hence a higher number of matches tend to be found among Dirty datasets.

Dataset name	Deduplication/ Record Linkage	Real /Synth	Clean /Dirty	No. of attributes	No. of records	No. of matches
Restaurant	Deduplication	Real	Clean	5	864	112
Cora	Deduplication	Real	Dirty	4	1,295	17,184
Clean-Synth	Deduplication	Synth	Clean	10	10,000	2,000
Dirty-Synth	Deduplication	Synth	Dirty	9	10,000	26,692
DBLP-ACM	Record Linkage	Real	Clean	4	2,616+2,294	2,224
DBLP-Scholar	Record Linkage	Real	Dirty	4	2,616+64,263	5,347

Table 2: Characteristics of datasets used in the empirical study.

Dominance Graphs

As there are 6 different datasets and 3 different linkage techniques there are 18 dominance graphs to be generated. Each dominance graph contains 12 nodes each representing one of the twelve variants of the blocking methods. The results from the blocking evaluation technique (Figures 3 $\rightarrow$ 8) then determine which nodes have edges between them, how many and in which direction. Each dominance graph is used to select a blocking method for a dataset using a particular linkage technique. For example, the dominance graph used to select a blocking method for *Restaurant* using SVM linkage, would therefore contain all the edges for the other five datasets using SVM linkage. The non-dominated nodes are identified and the method(s) with the largest $Incoming - Outgoing$ number of edges value is selected. The results table of the evaluation technique can indicate how closely the selected methods perform in comparison to the optimal methods.

6. Results and Discussion

In our experiments we evaluated three different blocking techniques A_i [2], B_i [20], C_i [27], with i indicating the maximum key conjunction length used by each blocking method, and two linkage approaches: SVM and TF-IDF. All of the blocking methods were evaluated with the proposed $F_{Prec,Rec}$ [Harmonic mean of precision and recall] vs Time technique described in Section 4.1. In Tables 3 to 8 we present the blocking quality results (Reduction Ratio [RR], Pairs Completeness [PC], Harmonic mean of RR and PC [$F_{\text{RR,PC}}$]) of each blocking method for each dataset along with the post-linkage results ($F_{\text{Prec,Rec}}$) of the applied linkage techniques. In Tables 9 to 14 we present the respective end-to-end (i.e. blocking and linkage combined) run times for each blocking method and subsequent linkage technique combination for each dataset. Please note that because Perfect linkage is instantaneous, its time values equate to that of blocking only. The respective SVM and TF-IDF time values therefore consist of these blocking times plus any incurred linkage time by each linkage technique. In the following section we provide in depth analysis of the obtained results.

6.1. Assessing Blocking Methods with the Proposed Evaluation Technique

As discussed in Section 4.1 we use graphs (Figures 4 to 21) depicting scatter plots of $F_{\text{Prec,Rec}}$ versus overall end-to-end time to evaluate and compare different blocking methods for each dataset. Since each blocking method is one of three blocking scheme learning algorithms using conjunctions of specific lengths from 1 to 4 keys, there are typically 12 points per graph. The blocking scheme learning algorithms of [2, 20, 27] are represented using Ai, Bi and Ci respectively. The adjoining i=1$\rightarrow$4 value indicates the maximum conjunction length used by each blocking method. Methods with excessively high running time in comparison to others are deliberately omitted, thus some graphs contain fewer than 12 points.

From a quick glance at the results we can see that some blocking methods were seen to perform quite well for some datasets but poorly in others. For example, B3 is among the top performing for *Clean-Synth* in Figures 12 and 18 and for *DBLP-ACM* in Figures 14 and 20, but is arguably the lowest performing for *Restaurant* in Figure 4.

In Figures 4 to 21 the optimal blocking method only ever tends to be either B1, B2 or C1. However, each of these blocking methods may not necessarily be a good method for all cases. For example, C1 is the optimal blocking method in all 3 linkages cases for *Restaurant* (Figures 4, 10 and 16) but is among the lowest performing in 2 of the linkage cases for Clean-Synth (Figures 12 and 18), DBLP-ACM (Figures 14 and 20) and DBLP-Scholar (Figures 15 and 21).

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.999	0.950	0.974	0.974	0.902	0.928
A2	0.999	0.955	0.976	0.977	0.920	0.942
A3	0.999	0.945	0.971	0.972	0.910	0.932
A4	0.999	0.938	0.967	0.968	0.908	0.927
B1	0.999	0.923	0.960	0.960	0.900	0.923
B2	1.000	0.926	0.961	0.962	0.904	0.926
B3	1.000	0.884	0.938	0.938	0.888	0.908
B4	1.000	0.893	0.943	0.943	0.893	0.915
C1	0.999	0.994	0.996	0.997	0.922	0.943
C2	0.999	0.967	0.983	0.983	0.922	0.937
C3	0.999	0.936	0.966	0.967	0.910	0.935
C4	1.000	0.955	0.977	0.977	0.918	0.932

Table 3: Numerical results for *Restaurant* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1$\rightarrow$4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.969	0.912	0.939	0.954	0.814	0.835
A2	0.970	0.917	0.943	0.957	0.821	0.839
A3	0.970	0.917	0.943	0.957	0.821	0.839
A4	0.970	0.917	0.943	0.957	0.821	0.839
B1	0.937	0.936	0.937	0.967	0.824	0.842
B2	0.935	0.939	0.937	0.968	0.823	0.842
C1	0.914	0.935	0.925	0.966	0.822	0.841
C2	0.972	0.883	0.925	0.938	0.819	0.833
C3	0.916	0.937	0.926	0.968	0.823	0.841
C4	0.974	0.883	0.926	0.938	0.822	0.836

Table 4: Numerical results for *Cora* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1→4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.999	0.999	0.999	0.999	0.350	0.352
A2	1.000	0.997	0.998	0.999	0.350	0.352
A3	1.000	0.999	0.999	1.000	0.350	0.352
A4	1.000	0.999	0.999	1.000	0.350	0.352
B1	1.000	0.998	0.999	0.999	0.350	0.352
B2	1.000	1.000	1.000	1.000	0.487	0.489
B3	1.000	0.997	0.999	0.999	0.487	0.489
C1	0.995	1.000	0.997	1.000	0.350	0.352
C2	0.999	0.999	0.999	1.000	0.354	0.356
C3	1.000	1.000	1.000	1.000	0.400	0.402

Table 5: Numerical results for *Clean-Synth* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1→4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.999	1.000	1.000	1.000	1.000	1.000
A2	0.999	1.000	1.000	1.000	1.000	1.000
A3	1.000	1.000	1.000	1.000	1.000	1.000
A4	1.000	1.000	1.000	1.000	1.000	1.000
B1	0.999	1.000	1.000	1.000	1.000	1.000
B2	1.000	1.000	1.000	1.000	1.000	1.000
B3	0.999	1.000	1.000	1.000	1.000	1.000
B4	1.000	1.000	1.000	1.000	1.000	1.000
C1	0.983	1.000	0.991	1.000	1.000	1.000
C2	0.997	1.000	0.998	1.000	1.000	1.000
C3	0.999	1.000	1.000	1.000	1.000	1.000

Table 6: Numerical results for *Dirty-Synth* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1→4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.948	0.998	0.972	0.999	0.933	0.931
A2	0.949	0.998	0.973	0.999	0.935	0.932
A3	0.989	0.998	0.993	0.999	0.964	0.955
A4	0.989	0.998	0.993	0.999	0.964	0.955
B1	0.987	1.000	0.993	1.000	0.900	0.903
B2	0.998	0.998	0.998	0.999	0.964	0.956
B3	0.998	0.998	0.998	0.999	0.964	0.956
B4	0.998	0.998	0.998	0.999	0.964	0.956
C1	0.990	0.999	0.994	1.000	0.902	0.904
C2	0.997	0.998	0.998	0.999	0.909	0.908
C3	0.999	0.999	0.999	0.999	0.925	0.925
C4	0.998	0.998	0.998	0.999	0.927	0.925

Table 7: Numerical results for *DBLP-ACM* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1→4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Blocking			Linkage ($F_{\text{Prec,Rec}}$)		
Method	RR	PC	$F_{\text{RR,PC}}$	Perfect	SVM	TF-IDF
A1	0.989	0.997	0.993	0.999	0.496	0.801
A2	0.996	0.997	0.996	0.998	0.507	0.801
B1	0.998	0.993	0.996	0.997	0.553	0.804
B2	0.998	0.993	0.996	0.996	0.555	0.804
C1	0.990	0.999	0.994	0.999	0.459	0.800

Table 8: Numerical results for *DBLP-Scholar* where Ai, Bi and Ci are the blocking algorithms of [2, 20, 27] respectively. i=1→4 are maximum key conjunction lengths. $F_{\text{RR,PC}}$ is the harmonic mean of Reduction Ratio and Pairs Completeness and $F_{\text{Prec,Rec}}$ is the harmonic mean of Precision and Recall calculated post-linkage.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	2.8	3.0	3.0
A2	3.1	3.3	3.2
A3	3.3	3.5	3.4
A4	3.7	3.9	3.8
B1	2.6	2.8	2.8
B2	2.8	3.0	2.8
B3	4.0	4.2	4.1
B4	6.6	6.8	6.7
C1	2.7	2.9	2.8
C2	3.3	3.5	3.4
C3	11.1	11.3	11.3
C4	124.8	125.0	125.0

Table 9: Combined blocking and linkage times for *Restaurant* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	70.2	84.4	92.8
A2	73.1	86.4	93.8
A3	76.4	89.8	97.2
A4	80.7	94.2	101.6
B1	69.7	97.1	115.3
B2	71.0	99.5	116.8
C1	77.6	113.7	137.4
C2	79.7	91.9	99.2
C3	110.4	145.7	168.8
C4	432.3	444.1	451.1

Table 10: Combined blocking and linkage times for *Cora* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	8.5	11.2	21.9
A2	9.3	10.9	17.3
A3	10.2	11.0	13.7
A4	11.9	12.6	15.4
B1	7.9	9.3	14.9
B2	8.5	9.1	11.2
B3	17.0	18.1	24.3
C1	8.3	23.1	83.6
C2	11.5	14.5	25.3
C3	65.6	67.1	72.0

Table 11: Combined blocking and linkage times for *Clean-Synth* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	12.2	14.5	23.9
A2	13.4	15.8	25.2
A3	16.0	17.7	24.1
A4	17.8	19.4	25.9
B1	11.2	13.2	21.3
B2	11.6	13.2	19.7
B3	32.2	25.3	34.5
B4	151.5	155.1	168.7
C1	14.0	62.8	229.5
C2	16.2	25.2	55.9
C3	48.0	51.7	65.3

Table 12: Combined blocking and linkage times for *Dirty-Synth* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	25.2	69.8	141.8
A2	26.0	69.2	139.3
A3	24.1	35.9	54.6
A4	24.3	36.1	54.8
B1	23.6	36.9	56.3
B2	23.4	25.6	28.2
B3	23.8	26.0	28.7
B4	26.1	28.3	31.1
C1	24.0	32.9	45.1
C2	28.3	31.3	36.1
C3	37.0	39.1	43.2
C4	76.6	78.4	84.2

Table 13: Combined blocking and linkage times for *DBLP-ACM* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

	Time(seconds)		
Method	Blocking	SVM	TF-IDF
A1	108.3	434.4	761.2
A2	195.6	328.6	470.7
B1	88.3	134.8	185.8
B2	197.0	244.3	296.3
C1	109.2	409.3	683.1

Table 14: Combined blocking and linkage times for *DBLP-Scholar* where Ai, Bi and Ci are the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are maximum key conjunction lengths.

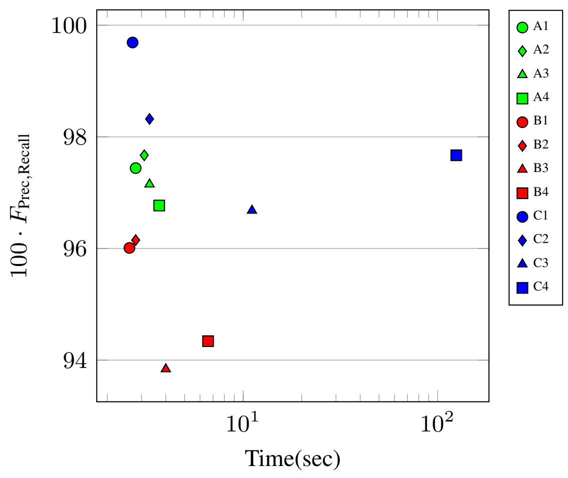


Figure 4: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Restaurant when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

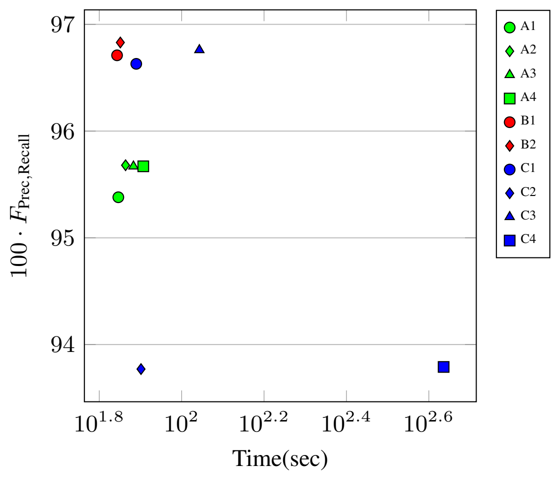


Figure 5: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Cora when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

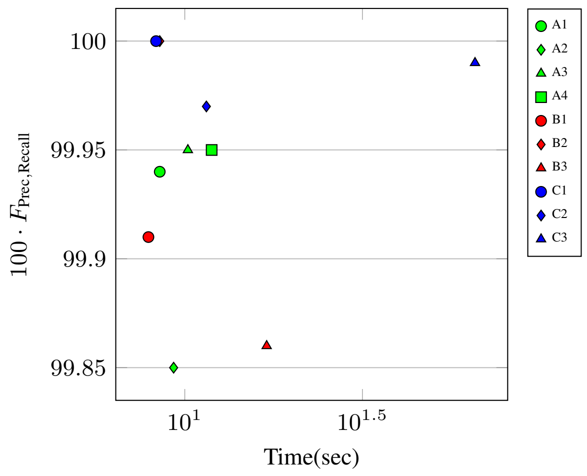


Figure 6: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Clean-Synth when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

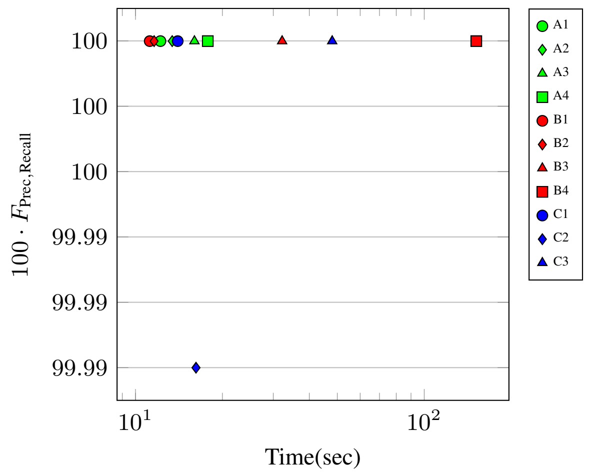


Figure 7: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Dirty-Synth when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

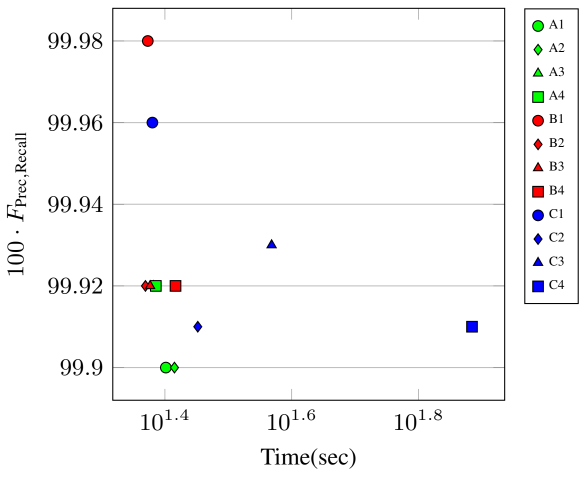


Figure 8: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-ACM when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

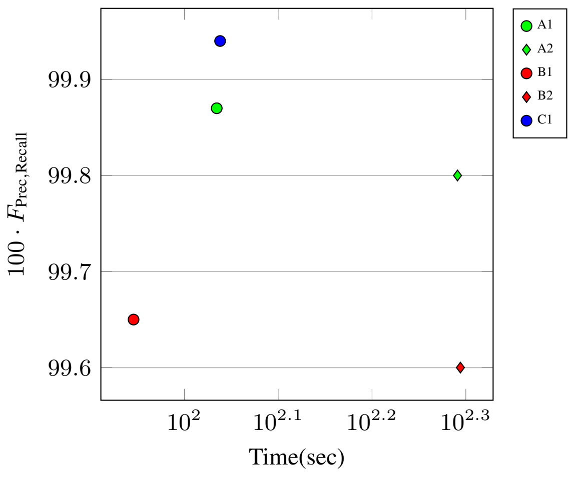


Figure 9: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-Scholar when using Perfect instantaneous linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

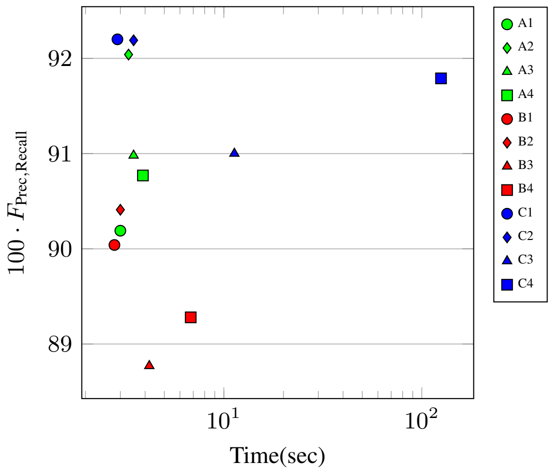


Figure 10: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Restaurant when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

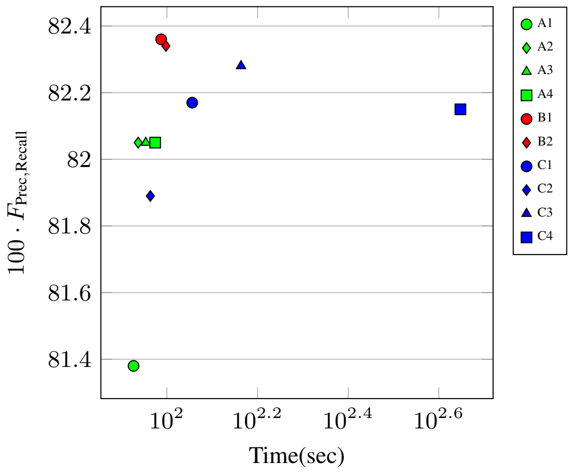


Figure 11: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Cora when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

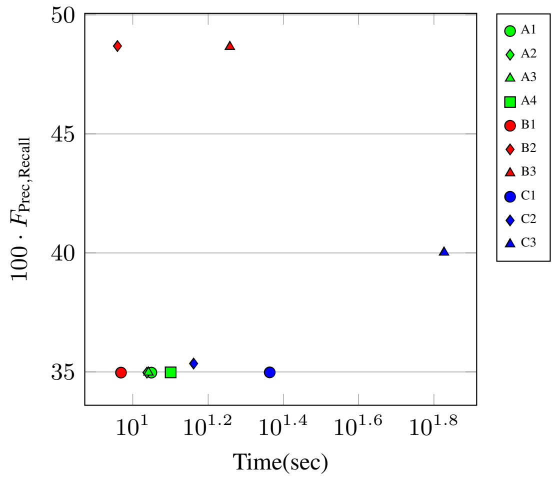


Figure 12: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Clean-Synth when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

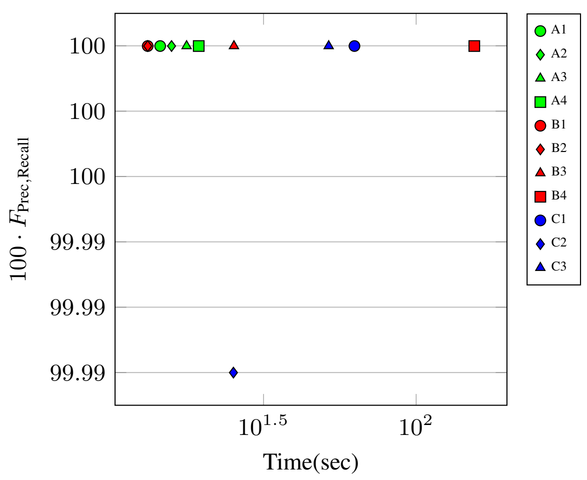


Figure 13: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Dirty-Synth when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

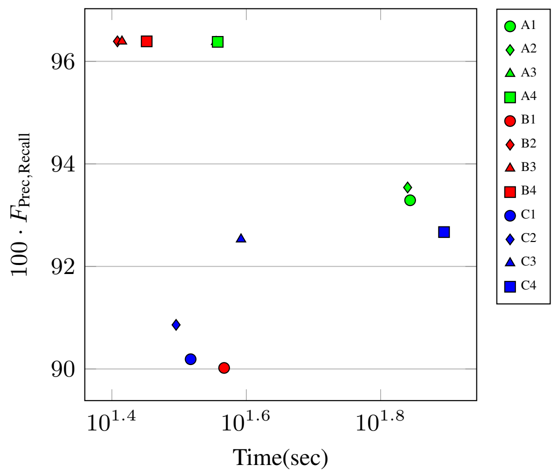


Figure 14: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-ACM when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

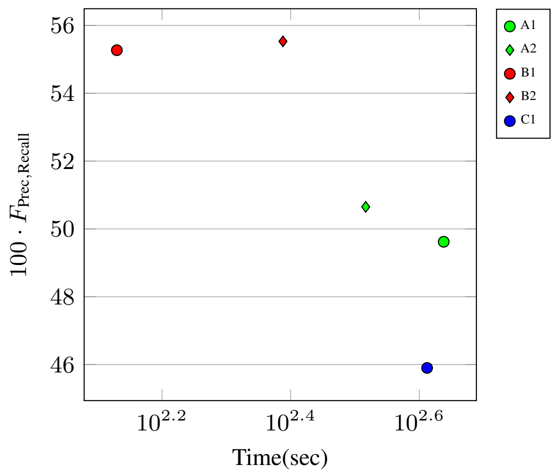


Figure 15: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-Scholar when using Support Vector Machine (SVM) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

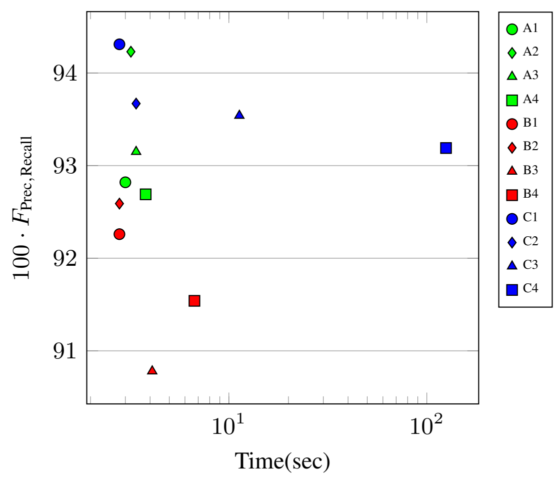


Figure 16: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Restaurant when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

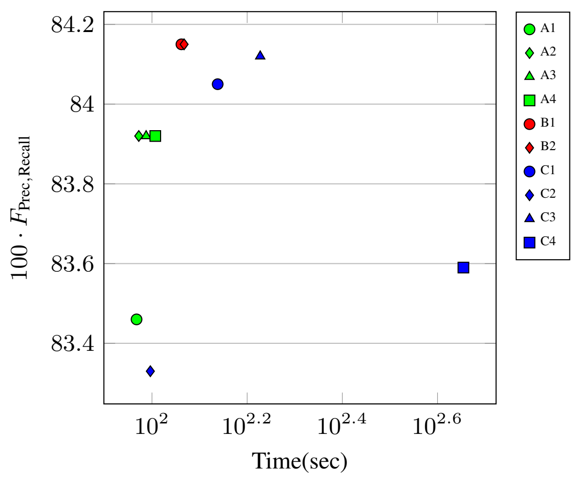


Figure 17: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Cora when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

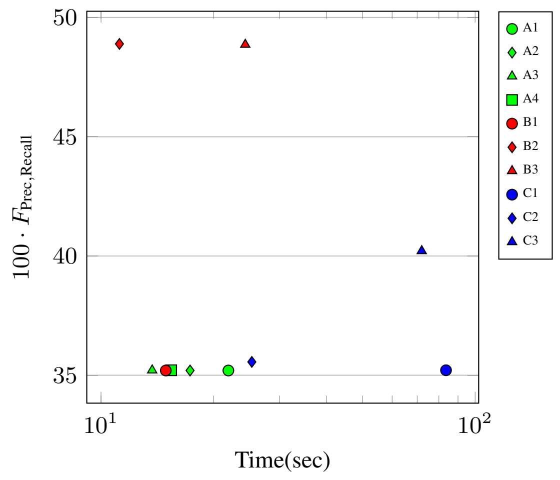


Figure 18: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Clean-Synth when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

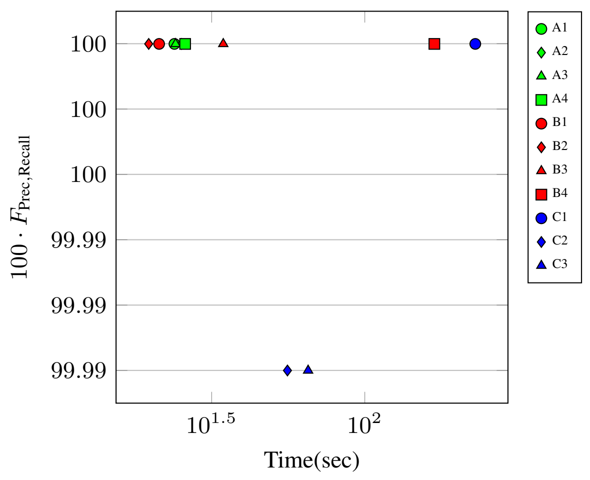


Figure 19: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for Dirty-Synth when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

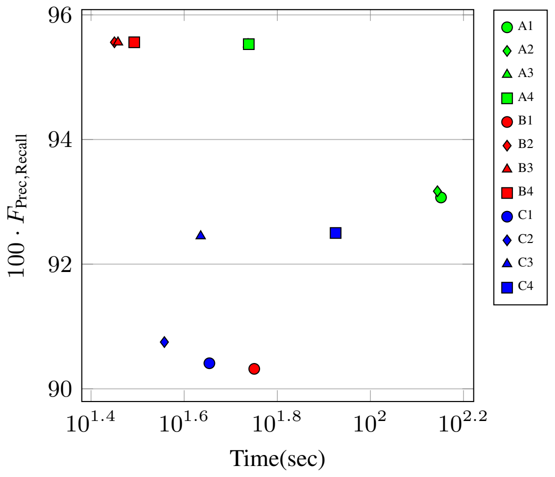


Figure 20: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-ACM when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

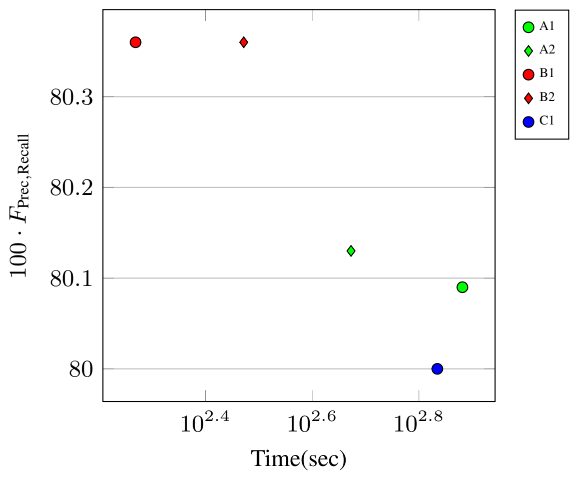


Figure 21: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall calculated post-linkage) vs Time graph for DBLP-Scholar when using Term Frequency-Inverse Document Frequency (TF-IDF) linkage. Where Ai, Bi and Ci represent the blocking scheme learning algorithms of [2, 20, 27] respectively and i=1→4 are the maximum key conjunction lengths used. If a method is considerably slower than the others it is not competitive and therefore omitted.

6.1.1. Blocking Method vs Linkage Technique

It is often observed that an optimal blocking method (assuming perfect linkage - a common assumption in the literature) is not necessarily an optimal method after an actual linkage takes place. From the results for dataset *DBLP-ACM* in Figures 8, 14 and 20 we can see that blocking methods perform differently depending on the linkage approach that is used later on. For example, in Figure 8 B1 and C1 are the optimal methods for *DBLP-ACM* if we assume the perfect linkage technique. However, in Figures 14 and 20 they are clearly outperformed by many other blocking methods when we apply either the SVM or TF-IDF linkage technique.

Metric	Perfect	SVM	TF-IDF
RR	No	No	No
PC	Yes	Yes	Yes
$F_{RR,PC}$	Yes	Yes	Yes

(a) Restaurant

Metric	Perfect	SVM	TF-IDF
RR	No	No	No
PC	Yes	No	Yes
$F_{RR,PC}$	No	No	No

(b) Cora

Metric	Perfect	SVM	TF-IDF
RR	No	No	No
PC	Yes	No	No
$F_{RR,PC}$	Yes	No	No

(c) Clean-Synth

Metric	Perfect	SVM	TF-IDF
RR	No	No	No
PC	Yes	Yes	Yes
$F_{RR,PC}$	Yes	Yes	Yes

(d) Dirty-Synth

Metric	Perfect	SVM	TF-IDF
RR	No	No	No
PC	Yes	No	No
$F_{RR,PC}$	No	No	No

(e) DBLP-ACM

Metric	Perfect	SVM	TF-IDF
RR	No	No	Yes
PC	Yes	No	No
$F_{RR,PC}$	No	No	No

(f) DBLP-Scholar

Table 15: Indication of whether the optimal blocking method(s) by the blocking evaluation metrics RR, PC or $F_{RR,PC}$ agree with the optimal method(s) by the linkage evaluation metric $F_{Prec,Rec}$ for each linkage technique and for each dataset.

The dataset *Clean-Synth* was deliberately generated to contain a significant number of similar but non-matching record pairs. This makes the task of classification especially challenging for all linkage techniques but the perfect linkage technique. This is most evident in Figure 6 where C1 achieves an $F_{Prec,Rec}$ measure of 1 upon *Clean-Synth* when assuming perfect instantaneous linkage, but is among the worse performing when considering SVM or TF-IDF linkage in Figures 12 and 18 respectively.

Blocking methods indicated as best by either RR, PC or $F_{RR,PC}$ are very often observed to be outperformed by others when evaluated as part of a RL framework i.e. when the results of a linkage technique are also considered. For example, in Table 7 C3 is indicated as the best blocking method for *DBLP-ACM* by $F_{\text{RR,PC}}$. However, when linkage is considered both the SVM and TF-IDF versions of B2 achieve superior $F_{Prec,Rec}$ results in less time than the respective versions of C3. In our experimental evaluations many similar cases were found, often with quite significant differences in $F_{\text{RR,PC}}$ pre-linkage. This also often applies if selecting the best blocking method by RR or PC. In Table 15, we show when the method(s) indicated as best by RR, PC or $F_{RR,PC}$ (pre-linkage) agree with the optimal method(s) by $F_{Prec,Rec}$ (post-linkage). PC and $F_{RR,PC}$ are observed to be better indicators of optimal blocking methods than RR but neither are correct every time. This confirms that blocking evaluation metrics alone cannot confidently indicate the best blocking method for a dataset, and that one must look to the post-linkage results as shown in Figures 4 to 21. Overall, these scatter graphs allow us to identify the optimal blocking method while taking into consideration both proficiency and resources (i.e. time). This is an important property when comparing methods, since it is arguably unfair to compare methods only by proficiency (regardless of which evaluation metric is used) if they are provided with different amount of resources (i.e. time).

6.1.2. Comparison of Individual Blocking Methods

The optimal blocking method of a $F_{Prec,Rec}$ vs Time graph was earlier defined as the point positioned in the top most left corner (Def 4.2). This is because it achieves the best balance of maximising $F_{Prec,Rec}$ and minimising Time in comparison to all other methods of the graph. In some cases one point may be clearly seen to achieve higher $F_{Prec,Rec}$ values in a lower runtime than all other points (See Figure 2a). For clarity we refer to such an optimal blocking method as a clear optimal blocking method. In our experimental evaluations a number of clear optimal blocking methods where observed, which provides a better picture about the optimal blocking methods available in the literature. Blocking method B1 is the clear optimal blocking method in 3 of the 7 figures in which it is an optimal method (Figures 7, 13 and 21). B2 is the clear optimal blocking method in 6 of the 7 figures in which it is an optimal method (Figures 12, 13, 14, 18, 19 and 20. C1 is the clear optimal blocking method in 1 of the 5 figures in which it is an optimal method (Figure 16). This suggests that B2 is more inclined than the other optimal blocking methods to be a clear optimal blocking method.

Blocking method C1 is indicated as the optimal blocking method for Restaurant by our evaluation technique regardless of which linkage technique is considered. The matching and non-matching record pair sets of Restaurant are the most highly separated of the evaluated datasets with an average TF-IDF similarity of 0.873, and 0.017, respectively. Furthermore, less than 0.0004% of the matching and non-matching record pair sets overlap by TF-IDF similarity. High linkage precision can be expected under these circumstances for all linkage techniques as classification is easy. In such cases a blocking method indicated as best by blocking evaluation metrics is likely to remain best post-linkage regardless of the choice of linkage technique. This suggestion is reinforced in Table 15a in which C1 is also indicated as the optimal method for Restaurant by the blocking evaluation metrics PC and $F_{RR,PC}$. Additionally, C1 is only 0.001 off the highest RR value achieved by any blocking method. We therefore observe that according to our proposed evaluation technique C1 is indicated as the best choice of blocking method for datasets in which the matches and non-matches are especially highly separated.

Looking at Tables 9 to 14 the computational runtimes of blocking methods Ai, Bi and Ci tend to increase as i=1$\rightarrow$4 increases. This increase only tends to be marginal for Ai and Bi in most datasets but is much more considerable for Ci. Despite this C1 and C2 were among the best performing in a number of cases in Figures 4 to 21, with C1 often being the optimal blocking method. This suggests that Ci (where i is maximum conjunction length) should be restricted to smaller values of i for large datasets of high dimension. This is most evident for the largest dataset DBLP-Scholar in Figure 21, where C2, C3 and C4 were omitted as their computational runtimes were considerably slower than others but C1 was the optimal blocking method overall. In conclusion, according to our proposed evaluation technique for relatively small and low dimension datasets blocking methods B3 and B4 should be avoided. For relatively large and high dimension datasets blocking methods Ai and Bi where i=1$\rightarrow$2 as well as C1 seem to be most appropriate.

For the larger datasets (Tables 11 to 14) we almost always see an initial decrease to end-to-end runtime for all blocking methods as i increases, before end-to-end runtimes begin to increase again. This is because although increasing i results in longer blocking runtimes, the linkage runtime typically reduces so dramatically that there is an overall improvement to the end-to-end runtime. For example, in Table 14 A2 has a higher blocking runtime, but lower end-to-end runtime, than A1 for DBLP-Scholar as it forms $\sim$1.2 Million fewer record pair comparisons. However, A3 and A4 are considerably slower than both A1 and A2 and thus omitted from the results. For blocking method Ai, the lowest end-to-end runtimes for the larger datasets are when i=2 or i=3 when using either linkage technique. For blocking methods Bi and Ci the lowest end-to-end runtimes are when i=2. This implies that although blocking method Ai may use conjunctions of up to i=3 for larger datasets, Bi and Ci should be restricted to at most i=2.

In summary, of the blocking methods evaluated using our proposed evaluation technique, analysis indicates that only B1, B2 or C1 are ever optimal (according to Definition 4.2). Of these B2 is nearly always a clear optimal blocking method indicating it as the strongest of the evaluated blocking methods. For datasets in which the sets of matching and non-matching record pairs are well separated the blocking method C1 is indicated as most suitable. We can also conclude that for relatively small and low dimension datasets the blocking methods B3 and B4 should be avoided. For relatively large and high dimension datasets Ai and Bi where i=1$\rightarrow$2 and C1 are considered as the most appropriate.

6.2. Selecting an Optimal Blocking Method for a New Dataset

In this section we present the results obtained by using our proposed blocking method selection technique based on the dominance graph described in Section 4.2. In Table 16 the methods selected by the proposed approach (Algorithm 1) are compared to the optimal blocking methods for each dataset using each linkage technique. It can be observed that the proposed approach selects an optimal or near-optimal method in every case. For the cases in which a non-optimal method is chosen, the $F_{Prec,Rec}$ and overall time values are still relatively close to that of the optimal method. It is also often the case when a non-optimal method is selected that although one of the values may be inferior, the other is superior, somewhat balancing the difference. For example, when selecting a method for DBLP-Scholar and using Perfect linkage, B1 is selected over the optimal method C1. However, although B1 has a lesser $F_{Prec,Rec}$ value than C1, it is considerably faster.

	Perfect		SVM		TF-IDF	
Dataset	Selected	Optimal	Selected	Optimal	Selected	Optimal
Restaurant	B1 0.960 2.6	C1 0.997 2.7	B2 0.904 3.0	C1 0.922 2.9	B2 0.926 2.8	C1 0.943 2.8
Cora	B1 0.967 69.7	B2 0.968 71.0	B2 0.823 99.5	B1 0.824 97.1	B2 0.842 116.8	B1 0.842 115.3
Clean-Synth	B1 0.999 7.9	C1 1.000 8.3	B2 0.487 9.1	B2 0.487 9.1	B2 0.489 11.2	B2 0.489 11.2
Dirty-Synth	B1 1.000 11.2	B1 1.000 11.2	B2 1.000 13.2	B1/2 1.000 13.2	B2 1.000 19.7	B2 1.000 19.7
DBLP-ACM	B1 1.000 23.6	B1 1.000 23.6	B2 0.964 25.6	B2 0.964 25.6	B2 0.956 28.2	B2 0.956 28.2
DBLP-Scholar	B1 0.997 88.3	C1 0.999 109.2	B2 0.555 244.3	B1 0.553 134.8	B2 0.804 296.3	B1 0.804 185.8

Table 16: $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall) and Time(seconds) values of the blocking methods selected by our proposed selection technique versus those of the optimal methods found from Figures 4 to 21 for each dataset and using each linkage technique. Where Bi and Ci represent the blocking scheme learning algorithms of [20, 27] respectively and i=1→2 are the maximum key conjunction lengths used.

In Figures 22 to 24 we compare the $F_{Prec,Rec}$ values of the selected methods against those of the optimal methods. As one can see, the selected methods are close to the optimal methods by $F_{Prec,Rec}$ in every case. Even in the very worst case, Restaurant using perfect linkage, the $F_{Prec,Rec}$ values only differ by 0.037. Looking at Table 16 we make a similar observation for the runtime values. In almost every case the selected methods have a time value close to that of the optimal methods. On average the selected methods are only ∼0.004% less proficient and ∼11.2 seconds slower than the optimal methods. Experimental evaluations indicate that although this selection approach cannot guarantee selection of an optimal method, it is highly likely that either an optimal or near optimal blocking method will be selected.

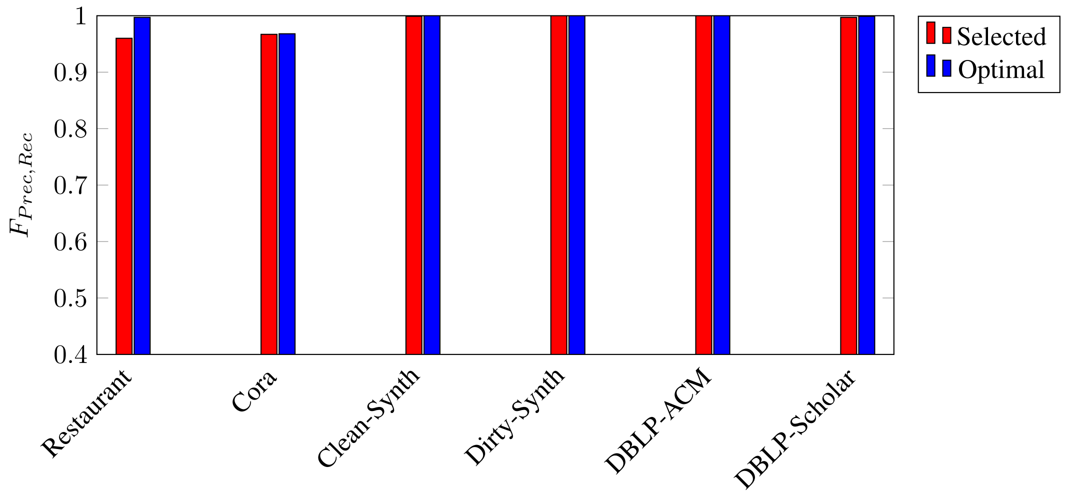


Figure 22: Comparison of $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall) values of selected and optimal methods when using perfect linkage.

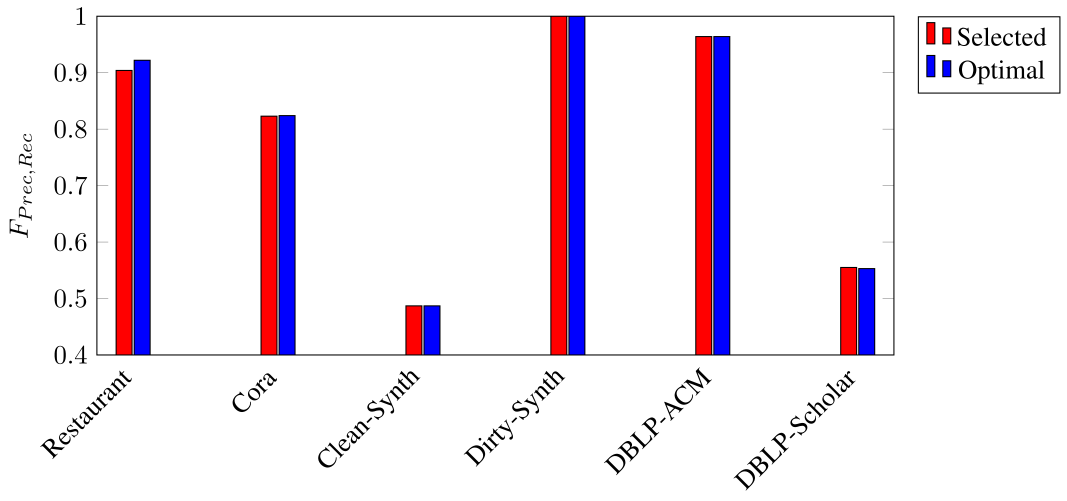


Figure 23: Comparison of $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall) values of selected and optimal methods when using SVM (Support Vector Machine) linkage.

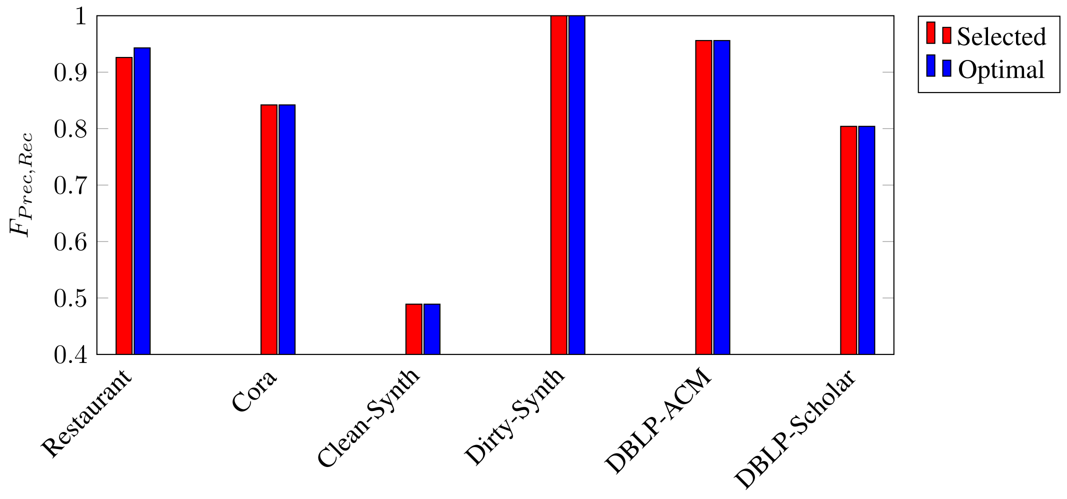


Figure 24: Comparison of $F_{Prec,Rec}$ (Harmonic mean of Precision and Recall) values of selected and optimal methods when using TF-IDF (Term Frequency-Inverse Document Frequency) linkage.

For all datasets, and considering all linkage techniques, the blocking methods Ai (where i=1→4) are never selected using our proposed technique nor indicated as optimal using our evaluation technique. Although this initially appears quite damning, by looking at Figures 4 to 21 we see that although these blocking methods are never optimal they are also rarely ever among the worse performing. By contrast the blocking methods Bi and Ci (where i=1→4) tend to achieve results in either extreme (i.e. among the best performing for some datasets and among the worse performing in others). This suggests that the blocking methods of Ai (where i=1→4) tend to be consistently ”safe” choices in comparison to Bi and Ci (where i=1→4). This highlights the effectiveness of the proposed selection technique being able to select an optimal or near-optimal method in every case considering the extreme temperament of some blocking methods.

In summary, our proposed selection technique always selects an optimal or near-optimal blocking method. In cases in which a near-optimal blocking method is selected one of the values is often superior to that of the optimal method. Blocking methods Bi and Ci (where i=1→4) tend to achieve either very good results or very poor results for different cases. By contrast blocking method Ai (where i=1→4) tends towards the centre of the cloud of points forming each $F_{Prec,Rec}$ vs Time graph, indicating it as a consistently safe choice.

7. Future Work and Conclusion

In this paper we present a complete comparison of different blocking methods. The experimental evaluation demonstrates that evaluation of blocking methods is a very challenging task. This is mainly due to the fact that different conclusions can be reached depending on what evaluation metric we apply and which subsequent linkage technique we use. We were able to present evidence to suggest that evaluation of blocking methods needs to take into consideration the subsequent linkage phase. It was shown that the optimal blocking method according to the currently used evaluation metrics does not necessarily guarantee an optimal RL framework when combined with different linkage methods. Evaluating blocking methods as part of RL frameworks with respect to overall proficiency and running time allows for a fairer comparison between approaches. We additionally observed when evaluating in this manner that the same blocking method achieved best results for our most well separated dataset regardless of which linkage technique was considered. In future work we would like to experiment with datasets with varying degrees of ambiguity to see to what extent is this a factor and if similar occurrences happen. Another observation when evaluating in this manner was that the more computationally demanding TF-IDF linkage technique tended to provide better post-linkage results (i.e. $F_{Prec,Rec}$) than the less computationally demanding SVM linkage technique in every dataset but one. Experimenting with a greater number of linkage techniques of varying complexity would make for an interesting area of future research. It would allow us to see how correlated the complexity and proficiency of different linkage techniques are and whether characteristics of a dataset can indicate a best linkage technique to use as part of a RL framework.

We also demonstrated the difficulty of selecting a good blocking method for previously unevaluated datasets using current blocking evaluation metrics. Multiple evaluations of different blocking methods can be timely to execute, assumes labelled training data to be available and does not take the subsequent linkage phase into consideration. Many blocking methods that achieved excellent results for some datasets were shown to work poorly for others. Indicating an uncertainty in their use upon previously unevaluated datasets. A blocking method selection approach was presented that overcomes these issues by using a conglomerate of results from the proposed evaluation technique on other datasets. This selection technique was shown to select an optimal or near optimal blocking method in every case.

In the future, we want to extend these experiments to substantially larger and varied datasets to verify whether the blocking method selection technique is further improved. We additionally wish to expand upon evaluating blocking methods as part of RL frameworks under time constraints, and use time constraints to automatically control the progress of the different phases of a RL framework. Time constraints could be used to determine at which point the blocking scheme learning process should stop and linkage should begin so that the entire RL process completes within a predefined time limit and achieves the best proficiency possible given the available resources.

References

[1] Bahmani, Z., Bertossi, L., Vasiloglou, N.: Erblox: Combining matching dependencies with machine learning for entity resolution. International Journal of Approximate Reasoning 83, 118–141 (2017)

[2] Bilenko, M., Kamath, B., Mooney, R.J.: Adaptive blocking: Learning to scale up record linkage. In: Data Mining, 2006. ICDM'06. Sixth International Conference on. pp. 87–96. IEEE (2006)

[3] Bilenko, M., Mooney, R., Cohen, W., Ravikumar, P., Fienberg, S.: Adaptive name matching in information integration. IEEE Intelligent Systems 18(5), 16–23 (2003)

[4] Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 39–48. ACM (2003)

[5] Bilenko, M., Mooney, R.J.: On evaluation and training-set construction for duplicate detection. In: Proceedings of the KDD-2003 Workshop on Data Cleaning, Record Linkage, and Object Consolidation. pp. 7–12 (2003)

[6] Bilenko, M.Y.: Learnable similarity functions and their application to record linkage and clustering. Ph.D. thesis (2006)

[7] de Carvalho, M.G., Laender, A.H., Goncalves, M.A., da Silva, A.S.: A genetic programming approach to record deduplication. IEEE Transactions on Knowledge and Data Engineering 24(3), 399–412 (2012)

[8] Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST) 2(3), 27 (2011)

[9] Christen, P.: A two-step classification approach to unsupervised record linkage. In: Proceedings of the sixth Australasian conference on Data mining and analytics-Volume 70. pp. 111–119. Australian Computer Society, Inc. (2007)

[10] Christen, P.: Automatic training example selection for scalable unsupervised record linkage. Advances in Knowledge Discovery and Data Mining pp. 511–518 (2008)

[11] Christen, P.: Febrl-: an open source data cleaning, deduplication and record linkage system with a graphical user interface. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1065–1068. ACM (2008)

[12] Christen, P.: Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection. Springer Science Business Media (2012)

[13] Dhillon, I.S., Ghosh, J.: Learnable similarity functions and their application to record linkage and clustering (2003)

[14] Duda, R.O., Hart, P.E., Stork, D.G., et al.: Pattern classification. 2nd. Edition. New York p. 55 (2001)

[15] Elfeky, M.G., Verykios, V.S., Elmagarmid, A.K.: Tailor: A record linkage toolbox. In: Data Engineering, 2002. Proceedings. 18th International Conference on. pp. 17–28. IEEE (2002)

[16] Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S.: Duplicate record detection: A survey. IEEE Transactions on knowledge and data engineering 19(1) (2007)

[17] Fisher, J., Christen, P., Wang, Q., Rahm, E.: A clustering-based framework to control block sizes for entity resolution. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 279–288. ACM (2015)

[18] Guillet, F., Hamilton, H.J.: Quality measures in data mining, vol. 43. Springer (2007)

[19] Isele, R., Jentzsch, A., Bizer, C.: Efficient multidimensional blocking for link discovery without losing recall. In: WebDB (2011)

[20] Kejriwal, M., Miranker, D.P.: An unsupervised algorithm for learning blocking schemes. In: Data Mining (ICDM), 2013 IEEE 13th International Conference on. pp. 340–349. IEEE (2013)

[21] Kejriwal, M., Miranker, D.P.: Semi-supervised instance matching using boosted classifiers. In: European Semantic Web Conference. pp. 388–402. Springer (2015)

[22] Kejriwal, M., Miranker, D.P.: An unsupervised instance matcher for schema-free rdf data. Web Semantics: Science, Services and Agents on the World Wide Web 35, 102–123 (2015)

[23] Kenig, B., Gal, A.: Mfiblocks: An effective blocking algorithm for entity resolution. Information Systems 38(6), 908–926 (2013)

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

[25] Köpcke, H., Thor, A., Rahm, E.: Evaluation of entity resolution approaches on real-world match problems. Proceedings of the VLDB Endowment 3(1-2), 484–493 (2010)

[26] Lehti, P., Fankhauser, P.: Unsupervised duplicate detection using sample non-duplicates. Lecture Notes in Computer Science 4244, 136 (2006)

[27] Michelson, M., Knoblock, C.A.: Learning blocking schemes for record linkage. In: AAAI. pp. 440–445 (2006)

[28] Papadakis, G., Ioannou, E., Palpanas, T., Niederee, C., Nejdl, W.: A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE Transactions on Knowledge and Data Engineering 25(12), 2665–2682 (2013)

[29] Papadakis, G., Papastefanatos, G., Koutrika, G.: Supervised meta-blocking. Proceedings of the VLDB Endowment 7(14), 1929–1940 (2014)

[30] Papadakis, G., Papastefanatos, G., Palpanas, T., Koubarakis, M.: Scaling entity resolution to large, heterogeneous data with enhanced meta-blocking. In: EDBT. pp. 221–232 (2016)

[31] Papadakis, G., Svirsky, J., Gal, A., Palpanas, T.: Comparative analysis of approximate blocking techniques for entity resolution. Proceedings of the VLDB Endowment 9(9), 684–695 (2016)

[32] Simonini, G., Bergamaschi, S., Jagadish, H.: Blast: a loosely schema-aware meta-blocking approach for entity resolution. Proceedings of the VLDB Endowment 9(12), 1173–1184 (2016)

[33] Whang, S.E., Menestrina, D., Koutrika, G., Theobald, M., Garcia-Molina, H.: Entity resolution with iterative blocking. In: Proceedings of the 2009 ACM SIGMOD International Conference on Management of data. pp. 219–232. ACM (2009)