Using Metric Space Indexing for Complete and Efficient Record Linkage

Özgür Akgün[1], Alan Dearle[1], Graham Kirby[1], and Peter Christen[2]

[1] School of Computer Science, University of St Andrews, St Andrews, Scotland. Contact: {ozgur.akgun, alan.dearle, graham.kirby}@st-andrews.ac.uk
[2] Research School of Computer Science, The Australian National University, Canberra, Australia. Contact: peter.christen@anu.edu.au

Abstract. Record linkage is the process of identifying records that refer to the same real-world entities in situations where entity identifiers are unavailable. Records are linked on the basis of similarity between common attributes, with every pair being classified as a link or non-link depending on their similarity. Linkage is usually performed in a three-step process: first, groups of similar candidate records are identified using indexing, then pairs within the same group are compared in more detail, and finally classified. Even state-of-the-art indexing techniques, such as locality sensitive hashing, have potential drawbacks. They may fail to group together some true matching records with high similarity, or they may group records with low similarity, leading to high computational overhead. We propose using *metric space indexing* (MSI) to perform *complete* linkage, resulting in a parameter-free process combining indexing, comparison and classification into a single step delivering complete and efficient record linkage. An evaluation on real-world data from several domains shows that linkage using MSI can yield better quality than current indexing techniques, with similar execution cost, without the need for domain knowledge or trial and error to configure the process.

Keywords: Entity resolution; data matching; similarity search; blocking.

1 Introduction

Record linkage, also known as entity resolution, data matching and duplicate detection [4], is the process of identifying and matching records that refer to the same real-world entities within or across datasets. The entities to be linked are often people (such as patients in hospital or customers in business datasets), but record linkage can also be applied to link consumer products or bibliographic records [4]. Record linkage is commonly challenged by the lack of unique entity identifiers (keys) in the datasets to be linked, which prevents the use of a database join. Instead, the linkage of records requires the comparison of the common attributes (or fields) that are available within the datasets, for example the names, addresses and dates of birth of individuals.

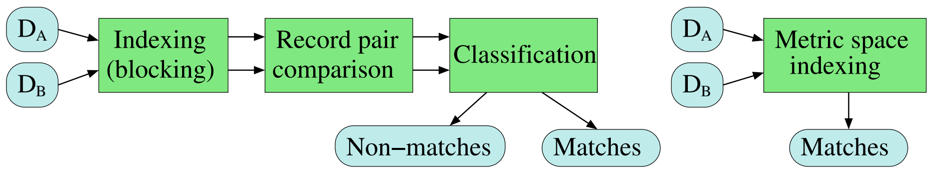


Fig. 1: Overview of the steps of the traditional record linkage process (left side) and our proposed metric space indexing based approach (right side), as described in Sect. 1, where records from two datasets, $\mathbf{D}_A$ and $\mathbf{D}_B$, are being linked.

To overcome data quality issues such as typographical errors and variations (common in name and address values [4]), approximate string comparison functions (e.g. edit distance, the Jaro-Winkler comparator, or Jaccard similarity [4]) are used to compare record pairs, leading to a vector of similarities (one similarity per attribute compared) for each pair. These are used to classify the record pairs into *links* (where it is assumed both records correspond to the same real-world entity) and *non-links* (where they are assumed to correspond to different entities). Various classification methods have been employed in record linkage [4,10], ranging from simple threshold-based to sophisticated clustering, supervised classification, and active learning approaches [30].

Besides a lack of unique entity identifiers, and data quality issues, linkage is also challenged by dataset scale [10]. To avoid full pair-wise comparison of all possible record pairs (quadratic in the dataset sizes), blocking techniques, commonly known as *indexing* [5], are used. These split the datasets into smaller blocks in an efficient way, grouping together records that are likely to correspond to the same entity. Only records within blocks are then compared in detail.

While indexing allows efficient linkage of large datasets [10], scalability is at the cost of reduced linkage quality, because potentially matching record pairs are ignored, leading to lower recall [4]. Indexing techniques, discussed in more detail later, range from simple phonetic based blocking [4] and sorting of the datasets [11] to locality sensitive hashing based techniques [18,29], and unsupervised [17,26] and supervised [1,22] learning of optimal blocking schemes.

Traditional linkage systems that perform indexing prior to comparison and classification (on the left in Fig. 1) add a further complexity. Indexing, comparison and classification are often conducted using algorithms and parameters selected using domain expertise, followed by manual assessment of the linkage outcomes [4]. If the resulting link quality is too low for a certain application, the process is repeated with different parameter settings or algorithms, giving a time-consuming iterative process [13]. The choice of an appropriate indexing technique as well as suitable parameter settings (including which attributes to use in indexing) will significantly affect the final linkage outcome.

We focus on approaches using a similarity threshold to classify links. These are fundamentally limited by the extent to which true matching records are similar, and true non-matches are dissimilar—this is dataset-dependent. Within this domain, we define a technique to be *complete* if it guarantees to find all record pairs within the specified threshold. Many indexing techniques are incomplete, since they reduce computational cost at the expense of potentially overlooking some true matches. By definition, incomplete techniques yield lower recall than complete ones. Conversely, and counter-intuitively, complete techniques can yield lower precision with some datasets. This is discussed further in Sect. 3.

Metric space indexing (MSI) is a complete technique with lower computational cost than a brute force approach. It allows indexing, comparison and classification to be combined into a single step (on the right in Fig. 1), making the process simpler, more efficient and more effective than incomplete approaches.

The motivation for this work is the Digitising Scotland project [9], which aims to transcribe and link all civil registration events recorded in Scotland between 1856 and 1973. This dataset will include around 14 million birth records, 11 million death records and 4 million marriage records.

Contribution: Our primary contribution is the novel application of MSI to achieve complete and efficient record linkage, without the need for complex parameter tuning. We evaluate our approach on several real-world datasets and demonstrate its advantages over existing indexing techniques for record linkage.

2 Related Work

We review relevant work in the areas of indexing for record linkage (for recent surveys see [5,25]), and metric space indexing [31]. Techniques to link records have been investigated for over five decades [12,24], with scalability being an ongoing challenge as datasets grow in size and complexity. Traditional blocking [5] uses a set of attributes (a *blocking key*) to insert records with the same value(s) in their blocking key into the same block. Only records within the same block are compared to each other. To overcome variations and misspellings, the values can be phonetically encoded using functions such as Soundex, NYSIIS, or Double-Metaphone [4]. These convert a string into a code according to its pronunciation, assigning the same code to similar sounding names (such as 'Gail' and 'Gayle'). Multiple blocking keys may also be used to deal with missing attribute values.

A different approach uses sorted neighbourhoods [23], where the datasets are sorted according to a *sorting key* (usually a concatenation of several attribute values). A sliding window is moved over the datasets and only records within the window are compared. Techniques that adaptively shrink or expand the window size based on the characteristics of the sorting key values have been shown to improve both linkage efficiency and quality [11].

These techniques are heuristics, requiring domain knowledge, such as the choice of appropriate blocking or sorting keys. Poor choices of blocking attributes result in records being inserted into inappropriate blocks, and thus true matches being missed, giving *incomplete* linkage. Conversely, many pairs compared in a block may have low similarity, being non-matches, giving *inefficient* linkage.

Locality sensitive hashing (LSH), proposed for efficient nearest-neighbour search in high-dimensional spaces [16], has been used for record linkage indexing. Attribute values are hashed multiple times, and blocks are created from those records that share some hash values. *HARRA* [18] is a linkage approach based on MinHash [3] and LSH which blocks, compares, and then merges linked records iteratively. [29] evaluates two LSH variations, concluding that to get good results, they must be tuned to the datasets. This requires good ground truth data which may be unavailable in real-world applications or expensive to obtain.

Metric space indexing (MSI) techniques [31] support similarity search. They require a distance measure between records, with certain properties including the *triangle inequality* [31]. Similarity search operations include *range-search(*$\boldsymbol{q}$*,*d*)*, identifying all records within a distance d of a query record $\mathbf{q}$; *nearest-neighbour(*$\boldsymbol{q}$*)*, returning the record with smallest distance to $\mathbf{q}$; and *nearest-n(*$\boldsymbol{q}$*,*n*)*, returning the n closest records to $\mathbf{q}$. Here we choose one MSI structure, the M-tree [6], and investigate its efficacy for record linkage. The M-tree is dynamically balanced. Every node contains a reference to a record being indexed, a pointer to its parent, the distance to its parent, and the node's radius. The radius of a node is the distance from it to its furthest child. For a parent node with radius r, all its children may be visualised as being contained within a ball of radius r from it.

A linkage method using R-trees [15] was described in [20], demonstrating that high linkage quality can be achieved using Jaccard similarity. [6] shows that M-trees are almost always more efficient than R-trees, hence their use here.

3 Approach

We address the following general linkage problem: for two datasets $\mathbf{D}_A$ and $\mathbf{D}_B$, we wish to find, for each record in $\mathbf{D}_A$, all the records in $\mathbf{D}_B$ that match it with regard to a certain distance threshold d (i.e. have a distance of d or less). We compare several linkage algorithms: traditional blocking, an incomplete similarity search method, LSH-MinHash, and a complete method, M-tree. We also use a complete brute force technique as a baseline, though this can only feasibly be applied to our smallest dataset. All experiments have a number of parameters to configure the search space and algorithm behaviour, including the distance function and the threshold, d, specifying the maximum distance for two records to be classified as a link (i.e. referring to the same entity). We focus on a single distance function in these experiments, to constrain the experimental space. In Sect. 5 we return to the selection of alternative distance functions.

Brute force: Every record in $\mathbf{D}_A$ is compared with every record in $\mathbf{D}_B$. Each pair is classified as a link if the distance between the records is less than or equal to the threshold d. This always finds all links, with complexity $O(|\mathbf{D}_A| \cdot |\mathbf{D}_B|)$.

Traditional blocking: The parameters are the set of blocking keys and (optionally) the phonetic encodings applied to each attribute. These are selected as described in [5], exploiting knowledge of the domain and of the data, and chosen with the intention of giving the best possible results. Each record in $\mathbf{D}_A$ is placed into the appropriate block based on its blocking key value. The algorithm then iterates over the records in $\mathbf{D}_B$, and for each one compares it with each of the records from $\mathbf{D}_A$ in the block with the same blocking key value.

Table 1: Characteristics of datasets used in the experiments.

Dataset name(s)	Records in dataset $\mathbf{D}_A$	Records in dataset $\mathbf{D}_B$	Number of true matching pairs	Entities linked
Cora	1,295	1,295	17,184	Publication–Publication
Isle of Skye	17,612	12,284	2,900	Birth–Death
Kilmarnock	38,430	23,714	8,300	Birth–Death

LSH-MinHash: The parameters for LSH-Minhash are [3] *shingle size* (l_{ss}), *band size* (l_{bs}) and *number of bands* (l_{nb}). First, the attributes of each record in $\mathbf{D}_A$ are concatenated, and the result *shingled* into a set of n-grams with $n = l_{ss}$. Next, a set of deterministically generated hash functions is applied to each n-gram in the set and the smallest result (the MinHash) of each hash application is added to a signature for the record. The number of hashes used, and thus the size of the signature, is set to $l_{nb} \times l_{bs}$. Finally, the signature is split into l_{nb} bands and the values from each band are hashed again to create a number of keys. The original record is added to a map associated with each of the keys. To perform linkage, the algorithm iterates over the records in $\mathbf{D}_B$. Each record is hashed as described above, to obtain a set of keys. Each key is looked up in the data structure, and the associated records from $\mathbf{D}_A$ added to the result set. Finally, the record from $\mathbf{D}_B$ is compared in turn with each record in the result set, with the pair being classified as a link or non-link based on their distance.

In some circumstances, incomplete approaches such as traditional blocking and LSH-MinHash can yield higher precision than complete techniques. This can occur when a significant number of non-matches nonetheless have high similarity. In this situation, the fact that an incomplete technique omits consideration of some potential links can serve to improve precision, since a classification decision based on a certain similarity threshold is incorrect for high-similarity non-matches. By definition, recall can never be higher for incomplete techniques.

M-tree: The linkage algorithm has no additional parameters. As with *LSH-MinHash*, each record in $\mathbf{D}_A$ is inserted into an M-tree. To perform linkage, the algorithm iterates over each record $\mathbf{b} \in \mathbf{D}_B$. A *range-search(**b**,d)* operation is performed on the M-tree, passing the distance threshold d as the second parameter. All the returned records are directly classified as links.

4 Experiments and Results

We now describe the datasets and method used in our evaluation [3]. We used three datasets from two domains in our experiments, as summarised in Table 1. The first is *Cora* [21], which contains 1,295 records that refer to 112 machine learning publications. Cora is commonly used as a benchmark dataset in the literature for assessing linkage algorithms. Ground truth is provided via a unique *paper_id*

[3] Experimental data, additional figures and source code can be downloaded from: `http://github.com/digitisingscotland/pakdd2018-metric-linkage`.

identifier of the form “blum1993”. In this experiment linkage is performed over the same set of records (i.e. a de-duplication [4]).

The other two datasets are historical Scottish records of vital events (birth, marriages and deaths), one registered on the *Isle of Skye*, a rural district, and the other records from *Kilmarnock*, an industrial town. These datasets were created, curated and linked by historical demographers [27,28]. Both include the names and genders of individuals and their parents. Ground truth was generated by the demographers based on their extensive domain knowledge.

In all of our experiments we use a single distance metric: the sum of the attribute-level Levenshtein [19] edit distances.

4.1 Cora Results

We perform linkage on the Cora dataset using all approaches presented in this paper: brute force, traditional blocking, LSH and M-tree, using several selected configurations for blocking and LSH. The distance threshold is varied between 0 and 250 [4]. For traditional blocking, the following attributes are used individually as blocking keys: *author*, *title*, *venue*, *location*, *publisher* and *year*. We also use a combined blocking key comprising all attributes.

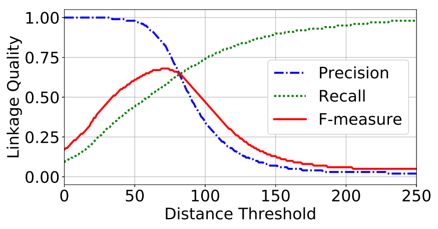
Figure 2 shows the precision, recall, and F-measure [4] for various thresholds [5]. As expected, low thresholds give high precision and low recall, and the reverse for high thresholds. Brute force and M-tree give identical results, as expected. The best linkage quality, with an F-measure of around 0.7, is achieved by several linkers, including brute force, M-tree, blocking on *authors*, blocking on all attributes, and two of the LSH configurations. All of these give similar overall results, apart from blocking on *authors*, which gives much better quality at very high distance thresholds. This is due to the incomplete nature of the approach, avoiding comparisons of significant numbers of high-similarity non-matches and thus avoiding these becoming false positives and keeping precision high.

For a more detailed investigation of selected linkers, the brute force approach is used to establish a good threshold value for the Cora dataset. The maximum F-measure is observed at a threshold value of $d = 70$. This value is dataset-dependent; for different datasets the maximum F-measure will occur at different thresholds. In the rest of this section we fix the threshold value at $d = 70$.

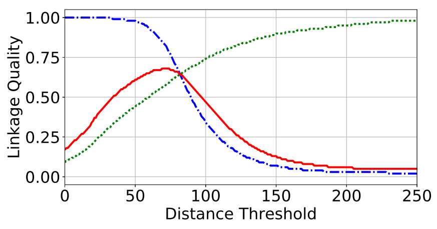
Table 2 shows greater detail for selected linkers, showing the parameters for the experiment, the number of distance comparisons made, and the precision, recall and F-measure achieved by each algorithm. In the *Linker* column the algorithm name is followed by its parameters: for LSH the number of the bands followed by the band size, and for traditional blocking the attributes used for blocking. The number of distance comparisons is reported as a machine-independent proxy for execution cost, since code profiling shows that distance calculations are dominant.

[4] Relatively high Levenshtein edit distances are included since Cora contains a number of low-similarity true matches.

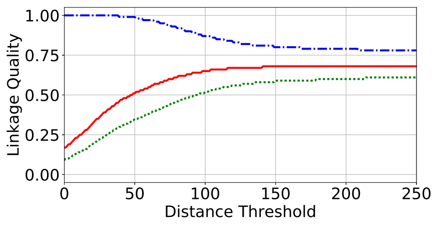
[5] Noting that recent research identifies some problematic aspects with using the F-measure to compare record linkage procedures at different similarity thresholds [14].



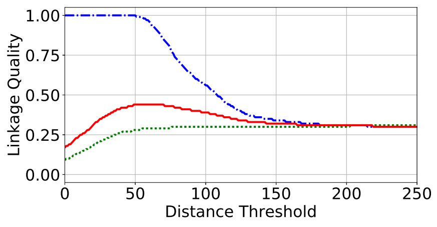
(a) Brute Force



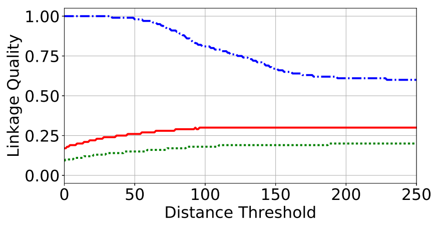
(b) M-tree



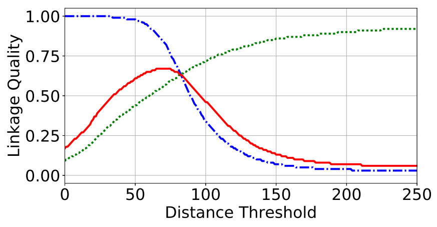
(c) Blocking on ‘authors’ attribute



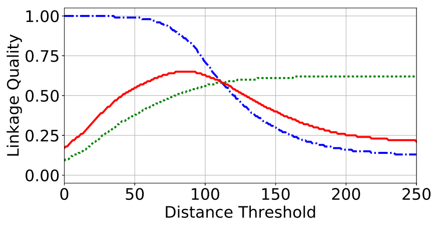
(d) Blocking on ‘title’ attribute



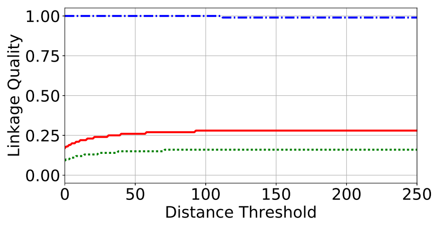
(e) Blocking on ‘publication year’ attr.



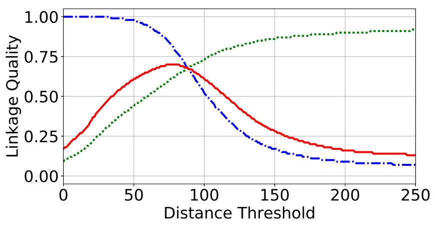
(f) Blocking using the union of all attrs.



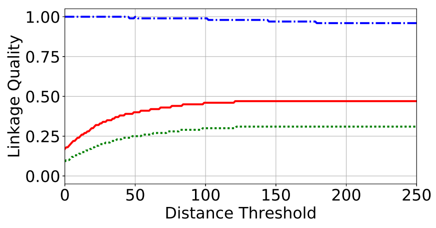
(g) LSH using 2 bands of size 2



(h) LSH using 2 bands of size 10



(i) LSH using 10 bands of size 2



(j) LSH using 10 bands of size 10

Fig. 2: Linkage results on the Cora dataset.

Table 2: Linkage quality on Cora dataset with distance threshold $d = 70$.

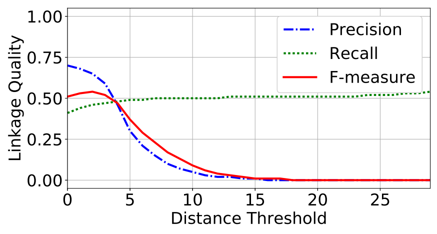
Linker	Comparisons	Precision	Recall	F-measure
Brute Force	1,677,025	0.84	0.57	0.68
M-tree	902,693	0.84	0.57	0.68
LSH-2-2	192,199	0.95	0.47	0.63
LSH-5-2	342,849	0.91	0.55	0.69
LSH-10-2	513,947	0.88	0.57	0.69
LSH-2-5	14,329	0.99	0.28	0.43
LSH-5-5	22,057	0.99	0.36	0.53
LSH-10-5	26,167	0.98	0.40	0.57
LSH-2-10	4,711	1.00	0.15	0.27
LSH-5-10	6,501	1.00	0.19	0.32
LSH-10-10	10,627	0.99	0.27	0.43
Block-year	115,893	0.99	0.35	0.51
Block-authors	11,039	0.94	0.16	0.28
Block-title	27,407	0.95	0.42	0.58
Block-venue	36,647	0.85	0.29	0.44
Block-location	1,009,957	0.83	0.43	0.57
Block-publisher	833,079	0.85	0.44	0.58
Block-combined	1,214,269	0.84	0.56	0.67

M-tree yields the same linkage quality as brute force, although using a significantly lower number of comparisons. This is as expected, since both techniques are complete. Several of the incomplete linkers give similar quality, for example *LSH-2-2*, *LSH-5-2*, *LSH-10-2* and *Block-combined*. These, and a number of other incomplete linkers, give better precision than the complete techniques. This is due to high-similarity non-matches, as discussed in Sect. 3. Although several of the incomplete linkers give as good quality as M-tree, and in some cases at lower cost, this is offset by the need to select appropriate configuration parameters. Some other linkers give very poor results.

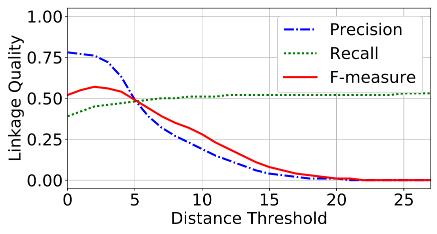
4.2 Demographic Dataset Results

Birth records were linked to death records, separately for the Skye and Kilmarnock datasets, using M-tree and a range of LSH configurations. It was not computationally feasible to run the brute force linker. Of the incomplete linkers, LSH was selected as it gave slightly better results for Cora. The shingle size was set to $l_{ss} = 2$ for all the LSH experiments reported, as this was found to give good results and LSH was not especially sensitive to this parameter. Results for other shingle sizes are omitted from this paper for brevity. A lower range of distance thresholds was explored, based on domain knowledge of the datasets.

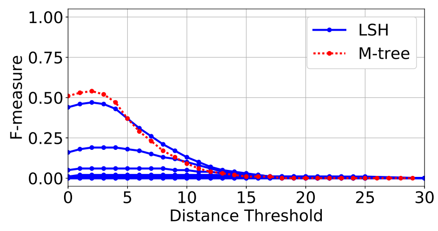
Figure 3 plots (a) and (b) show the M-tree precision, recall, and F-measure for various thresholds. In both datasets, the best F-measure values are obtained with



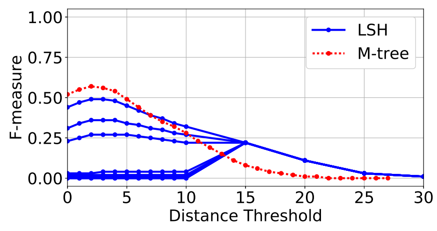
(a) M-tree on Isle of Skye dataset



(b) M-tree on Kilmarnock dataset



(c) F-measure on Isle of Skye dataset with M-tree and all LSH configurations



(d) F-measure on Kilmarnock dataset with M-tree and all LSH configurations

Fig. 3: Linkage results on the demographic datasets.

a low distance threshold of $d = 2$. Plots (c) and (d) compare the F-measure curves for M-tree with those obtained from a range of LSH configurations. The best F-measure value for M-tree is higher than that of any of the LSH configurations, for both datasets. This demonstrates both the competitiveness of M-tree with respect to linkage quality, and its important characteristic of being parameter-free—the linkage quality is obtained without the need to tune for the dataset.

Tables 3 and 4 show greater detail for selected linkers. In both cases the F-measure achieved is better for M-tree than any of the LSH linkers. The better linkage quality achieved by M-tree is largely due to recall for M-tree being much higher than for any of the LSH configurations. In most cases, LSH out-performs M-tree in terms of precision. More significantly, LSH linkage quality is heavily dependent on the configuration parameters. For plausible settings for the number of bands and band size, F-measure varies from 0.01 (extremely poor) to 0.47 (relatively good) for Skye and from 0.03 to 0.49 for Kilmarnock. In both cases *LSH-10-2* performs best, but since this is data-dependent there is no guarantee that these parameters would work well with another dataset.

The number of distance comparisons varies dramatically among the various linkers. M-tree always performs the most comparisons, since they are intrinsic to the *range-search* algorithm. The core part of the LSH linker performs Jaccard similarity comparisons and hashing; distance comparisons are only performed in the final step to determine whether a candidate pair is a link. The LSH configurations yielding the best results perform distance comparisons of the same order of magnitude as M-tree. This indicates that the good LSH linkers return many

Table 3: Linkage quality on Isle of Skye dataset with distance threshold $d = 2$.

Linker	Comparisons	Precision	Recall	F-measure
M-tree	102,318,525	0.65	0.46	0.54
LSH-2-2	3,109,250	0.63	0.03	0.06
LSH-5-2	10,412,496	0.64	0.11	0.19
LSH-10-2	53,874,127	0.68	0.36	0.47
LSH-5-5	36,566	0.76	0.01	0.01
LSH-10-5	129,873	0.72	0.01	0.02

Table 4: Linkage quality on Kilmarnock dataset with distance threshold $d = 2$.

Linker	Comparisons	Precision	Recall	F-measure
M-tree	514,871,153	0.76	0.45	0.57
LSH-2-2	99,145,887	0.81	0.16	0.27
LSH-5-2	130,721,338	0.79	0.23	0.36
LSH-10-2	177,168,848	0.79	0.36	0.49
LSH-5-5	239,368	0.84	0.01	0.02
LSH-10-5	855,431	0.87	0.02	0.03

candidates beyond the distance threshold. Thus, in order to get good results, LSH tends towards a brute force search over the candidate results. Despite this, LSH is faster due to the efficiency of the hashing process.

5 Conclusions and Future Work

In this paper we have demonstrated the efficacy of MSI in achieving complete and efficient record linkage, without the need for complex parameter tuning. In conclusion, this claim deserves some careful unpacking. It is always possible to achieve high quality linkage using a brute force approach. However the quadratic complexity of this approach prevents its practical application for datasets of even moderate size. We have shown that MSI techniques such as M-tree can deliver high precision, high recall results that are the same as those delivered by brute force. Furthermore this is achieved with fewer distance comparisons, and consistently without the need for complex parameter tuning.

We contrast this to traditional blocking and LSH-based approaches. Their major drawback is that whilst they can produce extremely good results, they can also produce extremely poor results. It was our observations of low recall given by these approaches that originally led us to experiment with M-trees.

We note that the good results obtained by both traditional blocking and LSH are partly due to the fact that (in the limit) they tend towards brute force as the number of records in the blocks increase. A second, unexpected, result is that illustrated in Table 2, namely that incomplete approaches such as LSH can in some cases yield higher precision than that achieved by a complete method such as M-tree. This is due to the incomplete linker masking the inability of a classifier based solely on record similarity to correctly classify high-similarity non-matches or low-similarity matches.

We have focused on a single distance function: the sum of attribute-level Levenshtein distances. This gives a straightforward intuition of record-level distances, but it is more common to normalise metrics to the range 0-1.

Many distance functions, such as Jaro-Winkler, are not metric, and therefore cannot be used with MSI techniques. Care must be taken to preserve metric properties when combining metrics over individual fields, as is highlighted in [2], since this may yield a function that is not metric. It is then possible for MSI techniques to yield results that are subtly incorrect.

Different metrics can give different distributions of inter-record distances, which can affect both the linkage results and the number of comparisons made, and hence the efficiency of the algorithm. Datasets with low variation in inter-record distances are said to have high *intrinsic dimensionality*, and tend to require high numbers of comparisons. The Scottish vital event datasets [28], combined with Levenshtein-based metrics, appear to have high intrinsic dimensionality. We are now, therefore, investigating the application of various different metrics to this domain, including Jensen-Shannon, Cosine and Structured Entropic Distance, as described in [7]. We are also investigating the applicability of a novel technique for dimensionality reduction [8].

6 Acknowledgements

This work was supported by ESRC grants ES/K00574X/2 "Digitising Scotland" and ES/L007487/1 "Administrative Data Research Centre—Scotland".

We thank Alice Reid of the University of Cambridge and her colleagues, especially Ros Davies and Eilidh Garrett, for the work undertaken on the Kilmarnock and Isle of Skye databases.

References

1. Bilenko, M., Kamath, B., Mooney, R.J.: Adaptive blocking: Learning to scale up record linkage. In: IEEE ICDM. pp. 87–96. Hong Kong (2006)
2. Bo, L., Yujian, L.: A normalized Levenshtein distance metric. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 1091–1095 (2007)
3. Broder, A.: On the resemblance and containment of documents. In: IEEE Compression and Complexity of Sequences. pp. 21–29. Salerno, Italy (1997)
4. Christen, P.: Data Matching – Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer (2012)
5. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE TKDE 24(9), 1537–1555 (2012)
6. Ciaccia, P., Patella, M., Rabitti, F., Zezula, P.: Indexing metric spaces with M-tree. In: Italian Symposium on Advanced Database Systems. vol. 97, pp. 67–86 (1997)

7. Connor, R.: A Tale of Four Metrics. In: Amsaleg, L., Houle, M.E., Schubert, E. (eds.) Similarity Search and Applications, pp. 210–217. Springer (2016)
8. Connor, R., Vadicamo, L., Rabitti, F.: High-Dimensional Simplexes for Supermetric Search. In: Similarity Search and Applications, pp. 96–109. Springer (2017)
9. Dibben, C., Williamson, L., Huang, Z.: Digitising Scotland (2012), `http://gtr.rcuk.ac.uk/projects?ref=ES/K00574X/2`
10. Dong, X.L., Srivastava, D.: Big data integration. Synthesis Lectures on Data Management 7(1), 1–198 (2015)
11. Draisbach, U., Naumann, F., Szott, S., Wonneberg, O.: Adaptive windows for duplicate detection. In: IEEE ICDE. pp. 1073–1083. Washington, DC (2012)
12. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association 64(328), 1183–1210 (1969)
13. Fisher, J., Wang, Q.: Unsupervised measuring of entity resolution consistency. In: IEEE ICDM DINA Workshop. pp. 218–221 (2015)
14. Hand, D., Christen, P.: A note on using the F-measure for evaluating record linkage algorithms. Statistics and Computing 28(3), 539–547 (2018)
15. Hjaltason, G.R., Samet, H.: Incremental distance join algorithms for spatial databases. SIGMOD Rec. 27(2), 237–248 (1998)
16. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: ACM TOC. pp. 604–613. Dallas (1998)
17. Kejriwal, M., Miranker, D.P.: An unsupervised algorithm for learning blocking schemes. In: IEEE ICDM. pp. 340–349. Dallas (2013)
18. Kim, H., Lee, D.: HARRA: fast iterative hashed record linkage for large-scale data collections. In: EDBT. pp. 525–536. Lausanne (2010)
19. Levenshtein, V.: Binary codes capable of correcting deletions, insertions and reversals. Cybernetics and Control Theory (10), 707–710 (1966)
20. Li, C., Jin, L., Mehrotra, S.: Supporting efficient record linkage for large data sets using mapping techniques. World Wide Web 9(4), 557–584 (2006)
21. McCallum, A.: Cora dataset: cora.csv (2017), `https://doi.org/10.3886/E4728V1`
22. Michelson, M., Knoblock, C.A.: Learning blocking schemes for record linkage. In: AAAI. Boston (2006)
23. Monge, A.E., Elkan, C.P.: The field-matching problem: Algorithm and applications. In: ACM SIGKDD. pp. 267–270. Portland (1996)
24. Newcombe, H., Kennedy, J., Axford, S., James, A.: Automatic linkage of vital records. Science 130(3381), 954–959 (1959)
25. Papadakis, G., Svirsky, J., Gal, A., Palpanas, T.: Comparative analysis of approximate blocking techniques for entity resolution. PVLDB 9(9), 684–695 (2016)
26. Ramadan, B., Christen, P.: Unsupervised blocking key selection for real-time entity resolution. In: PAKDD. Ho Chi Minh City (2015)
27. Reid, A., Garrett, E., Davies, R., Blaikie, A.: Scottish census enumerators' books: Skye, Kilmarnock, Rothiemay and Torthorwald, 1861–1901. Economic and Social Data Service (2006)
28. Reid, A., Davies, R., Garrett, E.: Nineteenth-century Scottish demography from linked censuses and civil registers: A 'sets of related individuals' approach. History and Computing 14(1-2), 61–86 (2002)
29. Steorts, R.C., Ventura, S.L., Sadinle, M., Fienberg, S.E.: A comparison of blocking methods for record linkage. In: PSD. pp. 253–268. Springer, Eivissa, Spain (2014)
30. Wang, Q., Vatsalan, D., Christen, P.: Efficient interactive training selection for large-scale entity resolution. In: PAKDD. Ho Chi Minh City (2015)
31. Zezula, P., Amato, G., Dohnal, V., Batko, M.: Similarity Search: The Metric Space Approach. Springer (2010)