Article: #1831-3044
Title: Learning Expressive Linkage Rules from Sparse Data
Authors: Petar Petrovski, Christian Bizer

Dear Reviewers,

thank you very much for the detailed and very helpful feedback on our article. We have improved the article based on your feedback and are now sending you a revised version together with point-by-point explanations on how we addressed each of your comments. The explanations are found below.

Best regards,
Christian Bizer and Petar Petrovski

Answers to the Reviewer's Comments

Review #1
Submitted by Anonymous
Recommendation: Major Revision

Comment R1C1: This restriction has also been considered in (Zhu et al., 2016) ([39] in the paper), where the authors use a graph summarization approach to address the problem of partly missing values for link discovery.
Answer R1C1: The authors of [39] (now [44]) were so nice to provide us with the source code implementing their approach. We ran the experiments from Section 5.3 using their code and now include the results of these experiments into the comparison. See results for CoSum-P in Table 5 and discussion in Section 5.3.

Comment R1C2: The authors then go on to present an appraisal of GenLink, pointing to its top performance above 95% F-measure. I do wonder whether this is necessary but if the authors want to keep this performance, they should point out on which datasets the performance is achieved as their performance claims contradict some of their later results.
Answer R1C2: Thank you for pointing out this impreciseness. We have toned down the claim about top performance and also made clear on which datasets GenLink achieved a performance above 95% F-measure. We now explicitly say that these datasets are all dense, which further motivates the argumentation line of the paper. The new sentence now reads: "The evaluation of GenLink on different dense datasets (e.g. sider-drugbank, LinkedMDB, restaurants) showed that the algorithm delivers good results with F-measures above 95\%[19]. As shown in the evaluation section of this paper, GenLink as well as other entity resolution methods run into problems once the data sets to be matched are not dense, but contain larger amounts of missing values."

Problem statement

Comment R1C3:- Notation: e_{a, b} = {(p_i, v_i)} is actually incorrect. The model should read "Each e \in A \cup B can be represented as a set of attribute-value pairs ...". Please correct.
Answer R1C3: This has been corrected in the paper.

Comment R1C4: - Notation: e_{a, b}: You run different indexes on p and v in your definition (e.g., p_1, v_2). Hence, (p_n, v_n) cannot be correct. Please fix.
Answer R1C4: Though, we do run separate indices for source and target entities, we don't run separate indices for property names and property values.

Comment R1C5: - "Find the subset M": What is M is subset of? Do you mean the set of pairs? Please specify.
Answer R1C5: Thank you. We do mean a subset of the available set of pairs. This has been clarified in the paper.

Comment R1C6:- Pairs of entities for which equality holds: e: e_a is not equal to e_b. owl:sameAs is an equivalence relation but not the equality relation. Please define clearly what you mean by equal (feel free to override the meaning thereof but please make sure it is formally sound).
Answer R1C6: As stated, by equality relation is considered to hold for entities which describe the same real world object(thing). The notation for the "equality relation" is abstracted to a similarity relation ~R.

Comment R1C7: Find its complement U. Isn't finding M equivalent to finding U as M = (A \times B) \setminus U?
Answer R1C7: Yes it is. This has been amended in the paper.

Comment R1C8: Definition of R+: The two resources e_a and e_b are not equal. Please fix.
Answer R1C8: This has been clarified in the paper.

Comment R1C9: Definition of R+: Formally the definition is unsound, as there is no guarantee that you can learn a ML model which generated M and U such that R+ \subseteq M and R- subseteq U. Please fix.
Answer R1C9: We do not provide any guarantees, we merely define the positive references as pairs of entities for which it is known that ~R holds. Given the definition of M, R+ has to be a part of M as well, therefore a subset.

Comment R1C10: Equation (3): You use * and \times to denote the cross products. Please stick to one (preferably \times).
Answer R1C10: This has been corrected in the paper.

Comment R1C11: - Linkage rule format: While the linkage rule description is appreciated, it still leaves space for interpretation. Please present a formal grammar for the linkage rules you use to ensure that the reader can reproduce your implementation if need be.
Answer R1C11: The formal grammar of the linkage rule language is presented in the original GenLink paper [1] which introduces this language (Page 1640, Section "Semantics"). We have added a reference to the paper to Section 3.1 making it clear that the formal grammar is found in this paper.

Comment R1C12: - The authors also claim that their linkage rule approach is "rather expressive" when compared to other grammars. What does this mean exactly? Can the author provide a formal interpretation of the statement?
Answer R1C12: The rules are considered expressive because they "subsume threshold-based boolean classifiers and linear classifiers and by that allows for representing non-linear rules and may include data transformations which normalize the values prior to comparison". This explanation been added to Section 3.1 of the paper. We prefer not to formalize the expressivity of the language, as we believe the verbal description will be more suitable for most readers of the article.

Comment R1C13: - Equation (5): Again, please stick to \times or *.
Answer R1C13: This has been amended in the paper.

Approaches

Comment R1C14: The authors claim that GenLink could perform well if one removed the penalty for long specifications. Please add this extension of the orginal algorithm to your evaluation.
Answer R1C14: We suggest that longer specifications have better chance of capturing nuanced differences between entities, however “removing (or loosening) the penalty has the potential to result with overfitted model and thus would not improve results”. This has been clarified on page 5 of the article. We also performed additional experiments in which we loosend and removed the penalty. Table 1 shows results of these experiments. The experiments indicate that the penalty only improves the results marginally for the WDC datasets.

	Headphones	Phones	TVs
	F-measure		
GenLink -- Penalty=0:05	0.799	0.712	0.748
GenLink -- Penalty=0.01	0.795	0.712	0.750
GenLink -- Penalty=0	0.795	0.709	0.715

Table 1. Results of the GenLink penalty removal experiment

Comment R1C15a: Group definition: The authors select the fittest individual. I guess they use the same initialization as with GenLink.
Answer R1C15a: The initialization is the same as GenLink: “The initial group is populated with the fitness individual from the population generated by GenLink.”

Comment R1C15b: Group fitness: Definition unclear. Please write an equation to clarify.
Do you mean you compute the union of the outputs of the groups and measure the fitness thereupon?
Answer R1C15b: Equations 6 and 7 have been added to the paper in order to explain the group fitness.

Comment R1C15c: Constant c: How did you establish the relation between c and the population?
Answer R1C15c: The relation is established based on the assumption that: “the larger the population, the bigger the chance for overfitting. Therefore, the constant should be higher for larger populations in order to penalise the fitness more.”

Comment R1C15d: Rule selection: The rules for matching are determined per pair. This means that all pairs from A \times B have to be compared. This is impractical for large datasets. Please report the runtimes of your algorithm in the experiments.
Answer R1C15d: In the group generation phase, we have to consider the whole reference sets (positive and negative) since we calculate the fitness from them.

In the group application phase the algorithm sets a rule per pair from a given set of unseen pairs. As of now the algorithm does A \times B comparisons, however any blocking approach can be applied to narrow the possible matchings. Silk, for instance, uses Miltiblock (http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/IseleJentzschBizer-WebDB2011.pdf) for reducing the number of comparisons on large datasets. In its current version, MultiBlock does not explicitly deals with sparse data, but it would be possible to adjust it using a similar penalty approach as GenLinkSA. As we have not yet implemented these ideas and not yet have a solid evaluation of them, we would prefer to leave this topic for a future publication.

Comment R1C16: GenLinkSA. The formal specification of GenLink aggregations is unclear. The sets S*, N* and F^a are not defined. The meaning of the arrow notation under the sets is also clear. Please rewrite or specify.
Answer R1C16: **"**The first argument S^* contains the similarity scores returned by the operators of this aggregation while the second argument N^* contains a weight for each of the operators, finally the third argument F^a represents the aggregation function that is applied to compute the similarity score S." This has been clarified in the paper.

Experiments

Comment R1C17: - All results were averaged: Please provide standard deviation values.
Moreover, given that you use genetic programming, please run statistical tests to state whether your average behaviour is significantly different from that of other algorithms. I would suggest running a rank test given that the experiments are ran 10x.
Answer R1C17: We fully agree with the reviewer on both points. We have added the standard deviation values to Table 6. We ran a Friedman nonparametric rank test [2] that confirms the results on the WDC dataset. We updated the paper accordingly.

Comment R1C18:- Interestingly, approaches such as (Nikolov et al., 2012) suggest that they need no training data to perform well on many of these datasets. So does (Zhu et al., 2016). Please compare with these approaches. Moreover, please consider the datasets in (Zhu et al., 2016).
Answer R1C18: We have extended the discussion of (Nikolov et al., 2012) and (Zhu et al., 2016) in the related work section. We have also added the results of CoSum-P (Zhu et al., 2016) to Table 9 covering the Amazon-Google dataset which as also used by (Zhu et al., 2016) and discuss the results on page 14.

Comment R1C19:- The authors compare with EAGLE, FEBRL and MARLIN. However, both algorithms have already been shown to be lacking in previous works. Please compare with more state-of-the-art approaches, e.g., Zhu et al. (2016) and Nikolov et al. (2012).
Answer R1C19: We run the CoSum-P approach as per Zhu et al. (2016) since the authors prove that it is better than the CoSum-B. We compare with this approach on the WDC dataset. Of the datasets used by Zhu et al. (2016), we chose the Amazon-Google dataset, as the F-measure for this dataset - 0.666 (see Table 6, pg. 13) shows it is the most challenging. As GenLink variants are a supervised approach, learning from examples improves on the performance of the CoSum-P. The results are confirmed by a Friedman test and a McNemar test.
The fitness function used by Nikolov et al. (2012) estimates precision and recall based on an assumption that "While different URIs are often used to denote the same entity in different repositories, distinct URIs within one dataset can be expected to denote distinct entities". This assumption is violated by many real world datasets. For instance, the WDC dataset contains many offers for the same product all originating from eBay. As this assumption does not fit the WDC dataset and as the authors do not publish an implementation of their approach (and also do not react to email asking for the implementation), we decided to just discuss Nikolov et al. (2012) in the related work section, but do not perform experiments using the WDC dataset.

Comment R1C20:- Why did you not consider removing the name property?
Answer R1C20: Within e-commerce data, the product name property often contains beside of the actual product name additional vendor-specific key product features, which might not be covered by the other properties in the datasets. In order to take advantage of this information, we keep the name property.
Comment R1C21: - How was the writing of the handwritten rules carried out?
Answer R1C21: The rules were written by the authors of the paper using their human knowledge about the respective products as well as statistics about the profile of the dataset. Figure 1 below shows as an example the handwritten rule that was used for matching headphones. The rule implements the idea that if the very sparse properties html:gtin or html:mpn match exactly, the record pair should be considered as a match. If the numbers do not match, the rule should fall back to averaging the similarity of the properties html:model, html:impedence and html:headphone_cup_type giving most weight to model. We updated the article and now explain on page 10 how the rules were created. In order not to extend the length of the article too much, we explain the idea behind the matching rule in the article, but do not include Figure 1 into the article which in our opinion does not add much over the verbal description of the idea.

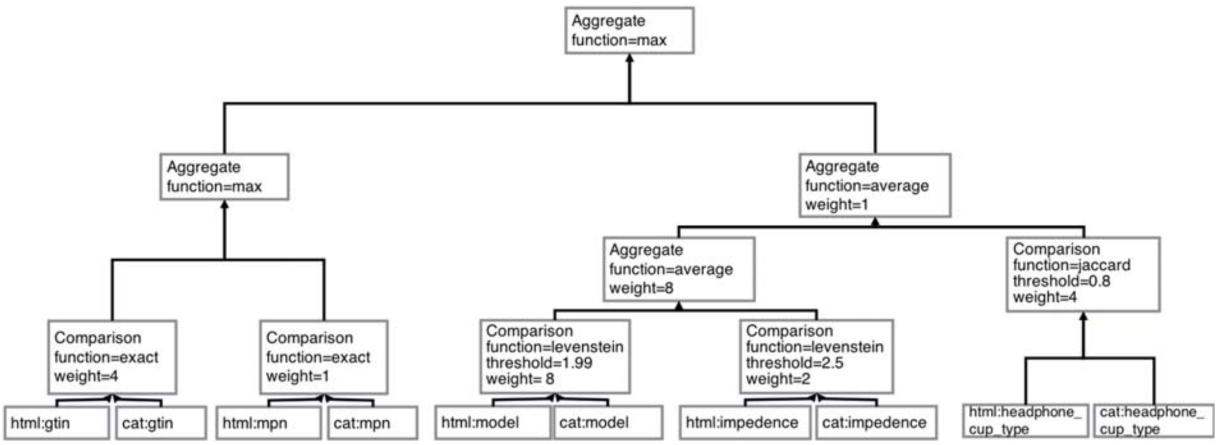


Figure 1. Example of a handwritten rule

Comment R1C22:- The authors claim that EAGLE has *significantly* lower results. How was significance measured?
Answer R1C22: The significance in the difference of the classifiers' accuracy was confirmed using the McNemar's test [3] with p<0.01.

Comment R1C23: - By a significant margin => Significance test?
Answer R1C23: We removed the significance claim here and now just say that headphones category proves to be an easier matching task. See page 13.

State of the art

Comment R1C24:- (Zhu et al., 2016) present an unsupervised approach for learning on RDF data. Please also consider the dataset they use.
Answer R1C24: We have extended the discussion of (Zhu et al., 2016 [44]) in the related work section where we again highlight the good performance of CoSum-P on the Amazon-Google dataset.

Comment R1C25:- (Nikolov et al., 2012) present a genetic unsupervised learning approach

Answer R1C25: A discussion of (Nikolov et al., 2012 [35]) has been added to the related work section.

References

Comment R1C26: One of the first works in the Semantic Web on this topic was by Nikolov et al., 2012. Please add it. Moreover, please add a reference to (Zhu et al., 2016) (i.e., [39]) here as it achieves a remarkable performance.
Answer R1C26: We added a discussion of both approaches to the related work section.

Comment R1C27:- Some references are unclear to me, e.g., Supervised [29]. Ngonga Ngomo et al., 2011 does not consider machine learning. This reference seems misplaced.
Answer R1C27: [29] (now [33]) Ngonga Ngomo et al : Raven - active learning of link specifications, 2011 introduces the RAVEN system that uses active learning to learn classifiers for link specifications.

Comment R1C28:- LIMES [29]. The LIMES framework is described in (Ngonga Ngomo, 2012). Please replace the reference.
Answer R1C28: The most cited reference for LIMES is [28] (now [32]): Ngomo, A.C.N., Auer, S.: Limes: A time-efficient approach for large- scale link discovery on the web of data 2011. This reference is also named in RAVEN [29] (now [33]) by Ngonga Ngomo et al. 2011 as the original publication covering the LIMES framework.

Comment R1C29:- Combination of the two [17]. Can you please explain how [17] does not fall into supervised approaches?
Answer R1C29: Thank you, this was miss referenced. The correct reference is: Isele, R., Jentzsch, A., Bizer, C.: Silk server-adding missing links while consuming linked data. In: Proceedings of the First International Conference on Consuming Linked Data.Volume 665. pp. 85–96 (2010). However, since we removed the mention of hybrid-approaches (see Answer R1C30) the reference was not added to the paper.

Comment R1C30:- How does one combine supervised and unsupervised approaches? The two paradigms are mutually exclusive.
Answer R1C30: With hybrid-approaches we meant approaches using semi-supervised learning. We removed the mention of hybrid-approaches from the paper as there are hardly any current systems in this category.

Comment R1C31:- Please cover active learning in the state of the art.
Answer R1C31: With RAVEN, Nikolov et al. and ActiveGenLink we refer to various systems in the related work section that use active learning. As active learning is not the focus of this article, we prefer not to add an extend discussion of active learning to the state of the art section. We also consider active learning of linkage rules on sparse data as a topic for further research and thus prefer to discuss it in a future paper once we have solid results on it.

Comment R1C32: Typos
- in the Linked Data => in the Linked Data Web?
- same real world object => same real-world object
- which is iteratively evolved => which is evolved iteratively
- gtin number => gtin
- footnotes after punctuation

- while still be able => while still being able
- low density properties => low-density properties
- top fitness individual => fittest individual
- thee => three
- Beside of comparing => In addition to comparing
- systems[5, 9] => systems [5, 9]
- there have been identified 112 duplicate records => 112 duplicate records were identified.
- main difficulty => main weakness
- "pivoting around" => rephrase, algorithms are not mobile
- MRLIN => MARLIN

Answer R1C32: All typos have been resolved. Thank you very much for pointing them out!

[1] Isele, R., Bizer, C.: Learning expressive linkage rules using genetic programming. Proceedings of the VLDB Endowment 5(11), 1638–1649 (2012)
[2] Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association 32(200), 675–701 (1937), https://www.tandfonline.com/doi/abs/10.1080/01621459.1937.10503522
[3] McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika 12(2), 153–157 (1947)

Review #2
Submitted by Anonymous
Recommendation: Minor Revision

Comment R2C1: The idea of searching for a family of rules is novel and intersting. The idea of adjusting the weights of the aggregation is intersting but simple, and not fully explored (eg, influence/sensitivity on beta parameter). The idea of combining the two approaches is original and interesting.
Overall, originality is good.
Answer R2C1: Thank you very much for your positive feedback on the originality of the approaches.

Comment R2C2: blocking: there is no mention of blocking, and it deserves at least some discussion as blocking is necessary for real world datasets, and blocking may also be affected by sparsity.
Answer R2C2: The linkage rules are learned only using the positive and negative training examples and thus blocking does not play a role for learning the linkage rules. As you say, blocking is crucial for applying the rules to larger datasets. Silk uses Miltiblock (http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/IseleJentzschBizer-WebDB2011.pdf) for reducing the number of comparisons on large datasets. In its current version, MultiBlock does not explicitly deals with sparse data, but it would be possible to adjust it using a similar penalty approach as GenLinkSA. As we have not yet implemented these ideas and not yet have a solid evaluation of them, we would prefer to leave this topic for a future publication.

Comment R2C3: How were the hyper-parameters tuned? Were they tuned for each dataset, or were the same setting used for all datasets? Tuned for each How were they tuned? What is the influence of the hyper-parameters on the results?
Answer R2C3: We thank the reviewer for this comment. The following explanation was inserted into Section 5.2. In order to address the comment: “Hyper parameters are set using grid search. Even though grid search was run for each dataset, the resulting parameter values were the same for all datasets. The

Table \ref{tab:avai_param} summarises the parameters that were used for GenLink and its variants in the experiments...”

Comment R2C4: learning times: there is no discussion of the learning times of the algorithms. This is important as genetic algorithms are very slow.
Answer R2C4: In order to address your comment, we have added a paragraph discussing the learning times and the used machine to Section 5.2. See beginning of page 11.

Comment R2C5: hand-written rules: the paper includes a vague reference for the rules being written by an expert. This is too vague as a determined user can write very good rules. In fact a determined user can write decision tree rules to deal with sparse values. Was this done? Would be good to list the hand written rule for one of the datasets.
Answer R2C5: We provide an example of one of the handwritten rules as part of the answer to Comment R1C21. In order to address your comment, we have added the following explanation to the article on page 10: “These rules are composed of up to six properties for each product category and were written by the authors of the article using their knowledge about the respective domains as well as statistics about the datasets. As an example, the handwritten rule that was used for matching headphones implements the idea that if the very sparse properties html:gtin or html:mpn match exactly, the record pair should be considered as a match. If the numbers are not present or do not match, the rule should fall back to averaging the similarity of the properties html:model, html:impedence and html:headphone_cup_type giving most weight to html:model.”

Comment R2C6: configuration of baselines and other systems: for replicability, the configuration of the other systems should be discussed and perhaps presented.
Was the default configuration used, was there an attempt to optimize it, eg, for FEBRL and EAGLE.
Answer R2C6: We used the default configurations for all systems. The parameter settings that were used for the baseline approaches are described in Section “5.2 Experiment Setup - Baselines”.

Comment R2C7: sparsification: random sparsification may unfarily hurt the machine learning (ML) approaches as as sparsity in real work datasets is not random and ML could identify it.
Answer R2C7: This is clearly an important point and also the reason why we devote more space to datasets that are naturally sparse (such as the WDC benchmark datasets) over making originally dense datasets sparse (e.g. restaurants, movies, and drugs), which we only present shortly in Section 5.4.

Comment R2C8: The paper mentions the importance of learning non-linear rules, and interestingly the main examples are linear, first example in fig 2 and fig 3, there is only one example of max
Answer R2C8: We have evaluated linear-rules versus non-linear rules in the initial GenLink paper (Isele/Bizer VLDB 2012, http://vldb.org/pvldb/vol5/p1638_robertisele_vldb2012.pdf). The results are found in Table 13 of this paper and show that the nonlinearity improves the performance by around 1% F1 in half of the cases. So nonlinearity is a useful feature, but clearly not the killer feature compared to other aspects of the method such a seeding or the specialized crossover operations (see Section 6.3 in Isele/Bizer VLDB 2012). That nonlinearity is also useful in some cases in the context of sparse data is shown by the GenLinkComb rules depicted in Table 8 of the article, which use quite a large number of min() operators.

Comment R2C9: Machine learning has developed many techniques to deal with missing values. Examples include methods that tolerate missing values and missing value imputation techniques that work well for numeric values and categorical variables with few categories.

Answer R2C9: We would consider the different extensions of GenLink presented in this article to be examples of methods that aim at tolerating missing values. Our experience with using imputation techniques in identity resolution settings is quite disappointing as the imputed values often seem to confuse the systems more than they help. This is clearly different for other classification tasks.

Comment R2C10: In traditional ML, a data scientist spends a significant amount of time developing features, selecting models and tuning hyper-parameters. The process leads to dramatic improvements over the first model that users try (eg, run SVM). A particular focus in this process is handling of missing values, through imputation, or by defining additional features (eg, a new indicator feature that specifies whether a value is present or not). While this process is not automated, a savvy data scientist can in a few hours achieve excellent results.
Answer R2C10: We believe the identity resolution is a area in which humans and machines ideally complement each other. While humans with their domain knowledge are good at feature engineering, value normalization and labeling corner cases, machines are much better at testing different similarity functions, setting thresholds, and assigning weights.

Comment R2C11: GenLinkSA uses a very simple formula to compute the score of a feature vector generated for a pair of records (using the beta parameter). It is likely that ML methods could learn a much better formula. This possibility makes me think about the significance of the contributions.
It is true that the simple algorithm presented in this paper improves the state of the art, but it also suggests that more sophisticated methods could do better. As this type of tuning is common in ML, it should be mentioned in the paper.
Answer R2C11: The question how much the additional expressivity of a deep neural network could improve the aggregation of feature vectors compared to the weighted nonlinear functions that we currently use is an interesting research question for future work. Referring again to the experiments that we presented in Section 6.3 of Isele/ Bizer VLDB 2012, the expressivity of the aggregation functions seams only one factor amongst others. Of course a ANN might find even (slightly?) better function, but the additional expressivity also raises the question whether there is enough labeled training data available within the specific use case in order to avoid overfitting. As labeling is often done by humans in data integration settings, training data is clearly an constrictive factor.

Comment R2C12: The paper is clearly written for the most part, but does contain a number of grammatical errors that should be cleaned up.
Answer R2C12: We did prove-read the paper and hope to have spotted all grammatical errors.

Comment R2C13: The GenLink overview in page 2 is somewhat vague. The authors refer the reader to the GenLink paper, but the overview should be precise. The crossover paragraph is too short and imprecise, eg "selects one operator at random in a parit of linkage rules" ... does this mean one operator in each rule? Do they have to be of the same kind (I suppose so). The next sentence talks about aggregation operators so it is not clear whether crossover applies to all 4 types of operators. A small amount of work can make the crossover section clear.
Answer R2C13: We thank the reviewer for improving the legibility of our paper. The section has been rewritten.

Comment R2C14: Group application: is the sorting of the rules done statically based on an analysis of the training data, or is the sorting of coverage done for each pair being tested? The examples suggests the second option, please clarify.

Answer R2C14: The sorting is done statically based on the coverage of reference set : “ The individuals in the input group are sorted by the percentage of coverage of the reference set. Sorting enables Algorithm 2 to find the more influential individual rules in less iterations.”

Comment R2C15: Equation 3 is complex and not explained.
Answer R2C15: Equation 3 is explained in the last paragraph of Section 2 and we have made the connection between the text and the equation more explicit: “The first argument *in the above formula* denotes a set of positive reference links, while the second argument denotes a set of negative reference links. The result of the learning algorithm is a linkage rule which should cover as many reference links as possible while generalising to unknown pairs.”

Comment R2C16: Intuition of equation 4 is not provided, a sentence would be enough.
Answer R2C16: The intuition is provided just before the formalism of the equation: “MCC [26] is defined as the degree of the correlation between the actual and predicted classes or formally”

Comment R2C17: Equations 6 and 7, although precise seem overly complicated to express a simple idea. As it stands, these equations are hard to follow.
Answer R2C17: We have changed the paper to provide a better explanation of the formalism of the aggregation operator (now equations 8 and 9): “The first argument S* contains the similarity scores returned by the operators of this aggregation while the second argument N* contains a weight for each of the operators, finally the third argument Fa represents the aggregation function that is applied to compute the similarity score S.” This has been clarified in the paper.

Review #3
Submitted by Oktie Hassanzadeh
Recommendation: Accept

Comment R3C1: (1) originality: this might be the weakest aspect of the work given that the algorithms are extensions of the authors' previous work. However, in my view, the quality of the solution, its impact, and the thorough experiments with very promising results makes this an original contribution to this field.
(2) significance of the results: The results are significant and have the potential to make a major impact on the state of the art in entity matching.
(3) quality of writing: The paper is very well written. It presents a hypothesis and evaluates it very well. I found it relatively easy to understand and follow, in part thanks to the good and real examples used throughout the paper. It is also positioned reasonably well in the literature.
Answer R3C1: Thank you very much for your positive feedback about our work!

Comment R3C2: First (this one needs to be fixed prior to acceptance), Results in Table 6 show you use "gtin", but it is not listed as one of the properties in Table 2.
Answer R3C2: Table 2 includes only properties that are filled in at least 10% of the records. The rule in Table 6 (now Table 7) that uses the gtin property has a coverage of 0.053 (see Column 4 of Table 7), meaning that it property is only filled in 5% of the cases. We added a note to the caption of Table 2 explaining that low density properties are not included into the table.

Comment R3C3: The only major flaw I see in the evaluation, which is related to the above issue, is lack of proper "ID" columns/properties in the data sets. Do you believe this is common in practice? You need to clarify the issue with "gtin" but were you able to retrieve any identifiers for the data sets? Perhaps they were used to build the ground truth?
Answer R3C3: The product descriptions that are contained in the WDC Gold standard(http://webdatacommons.org/productcorpus/index.html) were crawled in the first quarter of 2016. The statistics of the WDC November 2015 extraction (http://webdatacommons.org/structureddata/2015-11/stats/html-md.xlsx) show that at this time the gtin property was only used on 1.5% of the schema.org/Product pages. This number has risen a bit in the meantime, but the main problem of identity resolution for product data on the Web is still that only a small fraction of the websites provide product identifiers (I guess in order to make price comparisons harder).
We did not use the gtins to build the ground truth as the quality of the gtins found in web data is also rather low (e.g. some websites provide the same gtin number for all products they offer, likely due to errors in the script that renders their pages).

Comment R3C4: If possible, provide more details on how the ground truth labels are derived.
Answer R3C4: We manually generated 1500 positive correspondences, 500 for each product category. For each product of the product catalog at least one positive correspondence is included. Additionally, to make the matching task more realistic the annotators also annotate closely related products like: phone cases, TV wall mounts or headphone cables, ear-buds, etc. Furthermore we created additional negative correspondences exploiting transitive closure. Two independent annotators in parallel annotated the web pages. In case of a conflict, a third annotator solved them. More details on the dataset are published in referenced previous work [39].

Comment R3C5: Another issue I can see in the evaluation is that you do not have a baseline designed for the scenario you are addressing. You have baselines (TF-IDF or paragraph2vec) but they are the solutions one would use for dense data, not sparse data, as with all the other related work you compare with. One simple baseline you could try is the idea described in Section 5.6 of the following paper:
Hassanzadeh et al. Discovering Linkage Points over Web Data. PVLDB 6(6): 444-456 (2013) http://www.vldb.org/pvldb/vol6/p445-hassanzadeh.pdf
Basically, apply the linkage rules sequentially in the order of their expected "quality" ("strength"/coverage).
Could you also have a simpler adaptation of the original GenLink algorithm as baseline? One could also imagine a decision tree based approach.
Answer R3C5: Thank you for proposing this additional approach as well as the follow up conversation via email. We now discuss this approach in the related work [17]. Additionally, we implemented your idea, but found that applying the linkage rules sequentially in the order of their expected "quality" yield worse results than our two baselines, since only one pair (with highest similarity) of records of potentially one-to-many can be selected as match, so we did not include this experiments into the article. Additionally, we provide decision tree- and random forest-based approaches as further baselines in the evaluation section (table 5).

Comment R3C6: In Algorithm 2 description, you say Algorithm 2 takes input of algorithm 2 (you mean 1?).
Answer R3C6: Thank you, this is corrected in the paper.

Comment R3C7: Can you specify the details of the handwritten rules mentioned on page 11.

Can you show examples or link to a supplementary material?
Answer R3C7: We provide an example of one of the handwritten rules as part of the answer to Comment R1C21. In order to address your comment, we have added the following explanation to the article on page 10: “These rules are composed of up to six properties for each product category and were written by the authors of the article using their knowledge about the respective domains as well as statistics about the datasets. As an example, the handwritten rule that was used for matching headphones implements the idea that if the very sparse properties html:gtin or html:mpn match exactly, the record pair should be considered as a match. If the numbers are not present or do not match, the rule should fall back to averaging the similarity of the properties html:model, html:impedence and html:headphone_cup_type giving most weight to html:model.” In order not to extend the length of the article too much, we explain the idea behind the matching rule in the article, but do not include Figure 1 into the article which in our opinion does not add much over the verbal description of the idea.

Comment R3C8: A strong aspect of this paper is making available the source code, but the code has zero documentation. This is hardly any better than not making any code available, since someone not familiar with your code can hardly run it.
I strongly encourage you to write down at the very least a simple "getting started" guide that allows me to run, for example, your running example in the paper. This makes your paper truly repeatable, and allows me to apply it in my similar application scenarios (without having to contact you for further details!).
Answer R3C8: Thank you for this very useful suggestion. We have fully implemented it and now provide a wiki page [1, 2] where we explain how to run the running example of the paper, and link to the Silk framework’s wiki on how to prepare your own matching projects.

[1] https://github.com/petrovskip/silk.2.6-GenLinkSA/wiki
[2] https://github.com/petrovskip/silk.2.6-GenLinkGL/wiki

Semantic Web 0 (2016) 1–0
IOS Press

Learning Expressive Linkage Rules from Sparse Data

Petar Petrovski, Christian Bizer
Data and Web Science Group, University of Mannheim, B6, 26, 68159 Mannheim
E-mail: petar@informatik.uni-mannheim.de, chris@informatik.uni-mannheim.de

Abstract.
A central problem in the context of the Web of Data, as well as in data integration in general is to identify entities in different data sources that describe the same real-world object. There exists a large body of research on entity resolution. Interestingly, most of the existing research focuses on entity resolution on dense data, meaning data that does not contain too many missing values. This paper sets a different focus and explores learning expressive linkage rules from as well as applying these rules to sparse web data, i.e. data exhibiting a large amount of missing values. Such data is a common challenge in various application domains including e-commerce, online hotel booking, or online recruiting. We propose and compare three entity resolution methods that employ genetic programming to learn expressive linkage rules from sparse data. First, we introduce the *GenLinkGL* algorithm which learns groups of matching rules and applies specific rules out of these groups depending on which values are missing from a pair of records. Next, we propose *GenLinkSA*, which employs selective aggregation operators within rules. These operators exclude misleading similarity scores (which result from missing values) from the aggregations, but on the other hand also penalize the uncertainty that results from missing values. Finally, we introduce *GenLinkComb*, a method which combines the central ideas of the previous two into one integrated method. We evaluate all methods using six benchmark datasets: three of them are e-commerce product datasets, the other datasets describe restaurants, movies, and drugs. We show improvements of up to 16% F-measure compared to handwritten rules, on average 12% F-measure improvement compared to the original GenLink algorithm, 15% compared to EAGLE, 8% compared to FEBRL, and 5% compared to CoSum-P.

Keywords: Entity Resolution, Sparse Data, Linkage Rules, Genetic Programming, Link Discovery

1. Introduction

As companies move to integrate data from even larger numbers of internal and external data sources, more and more structured data is becoming available on the public Web. These data consist of HTML tables, as well as Linked Data and Microdata annotations. Establishing links between entities in different data sources that describe the same real-world object takes greater focus within even more application scenarios. There exists an extensive body of research in entity resolution. Additionally, a number of link discovery tools have been developed, which generate RDF links between entities in different data sets that represent the same real-world object. For example, several semi-automatic link discovery tools - such as Silk [19] or LIMES [32] - have been developed. These tools compare entities in different Linked Data sources based on user-provided linkage rules which specify the conditions that must hold true for two entities in order to be interlinked. However, most existing approaches focus on dense data [8, 19, 31, 32]. This paper sets an alternative focus and explores learning expressive linkage (matching) rules as well as applying these rules to sparse data, i.e. data that contains a large amount of missing values.

A prominent example of an application domain that involves data exhibiting lots of missing values is e-commerce. Matching product data from different websites (e.g. Amazon and eBay) is difficult as most websites publish heterogeneous product descriptions using proprietary schemata which vary widely concerning their level of detail [30]. For instance in [38], we analyzed product data from 32 popular e-shops. The shops use within each product category (mobile phones, headphones, TVs) approximately 30 different attributes to describe items. The subset of the attributes that are used depends on the e-shop and even on the specific product. This leaves a data aggregator that collects product data for many e-shops into a rich schema with lots of missing values.

In [19], we presented GenLink, a supervised learning algorithm that employs genetic programming in order to learn expressive linkage rules from a set of existing reference links. These rules consist of attribute-specific preprocessing operations, attribute-specific comparisons, linear and non-linear aggregations, as well as different weights and thresholds. The evaluation of GenLink on various dense datasets (e.g. sider-drugbank, LinkedMDB, restaurants) showed that the algorithm delivers good results with F-measure above 95% [19]. As shown in the evaluation section of this paper, GenLink as well as other entity resolution methods run into problems once the datasets to be matched are not dense, but contain larger amounts of missing values.

In order to overcome the challenge of missing values, this article introduces and evaluates three methods that build on the GenLink algorithm. First, we present *GenLink Group Learning (GenLinkGL)*, an approach that groups linkage rules based on product attribute diversity, thus successfully circumventing missing values. Next, we introduce the *GenLink Selective Aggregations (GenLinkSA)* algorithm which extends the original approach with selective aggregation operators to ignore and penalize comparisons that include missing values. Finally, we introduce *GenLinkComb*, an algorithm that combines the central ideas of the previous two into a integrated method. We evaluate all methods using six benchmark datasets: three of them are e-commerce product datasets, the other datasets describe restaurants, movies, and drugs.

The rest of this paper is structured as follows: Section 2 formally introduces the problem of entity resolution. Section 3 gives an overview of the GenLink algorithm. Subsequently, in Section 4 we introduce GenLinkGL, GenLinkSA and GenLinkComb methods for dealing with sparse data. Section 5 presents the results of the experimental evaluation in which we compare the proposed methods with various baselines as well as other entity resolution systems. Section 6 discusses the related work.

2. Problem Statement

We consider two datasets, A the source, and B the target dataset. Each entity $e \in A \cup B$ consists of a set of attribute-value pairs (properties) $e = \{(p_1, v_1), (p_1, v_2), \ldots, (p_n, v_n)\}$, where the attributes are numeric, categorical or free-text. For instance, an entity representing a product might be described by the name, UPC, color, camera properties as shown in Figure 1. Our goal is to learn a matching rule that determines whether a pair of entities (e_a, e_b) represents the same real-world object. Or formally, given the two datasets A and B, the objective is to find the subset M consisting of all pairs of entities for which a relation $\sim_R$ holds and is defined by [14]:

$$M = \{(e_a, e_b); e_a \sim_R \; e_b, e_a \in A, e_b \in B\} \qquad (1)$$

Additionally, find its complement subset U defined as:

$$U = (A \times B) \setminus M \qquad (2)$$

To infer a rule specifying the conditions which must hold true for a pair of entities to be part of M, we rely on a set of positive correspondences $R_+ \subseteq M$ that contains pairs of entities for which the $\sim_R$ relation is known to hold. Analogously, we rely on negative correspondences $R_- \subseteq U$ that contain pairs of entities for which the $\sim_R$ relation is known not to hold.

Given the correspondences, we can define the purpose of the learning algorithm as learning matching rules from a set of correspondences:

$$m : 2^{(A \times B)} \times 2^{(A \times B)} \rightarrow (A \times B \rightarrow [0, 1]) \qquad (3)$$

The first argument in the above formula denotes a set of positive reference links, while the second argument denotes a set of negative reference links. The result of the learning algorithm is a linkage rule which should cover as many reference links as possible while generalising to unknown pairs.

3. Preliminaries

GenLink is a supervised algorithm for learning expressive linkage rules for a given entity matching task.

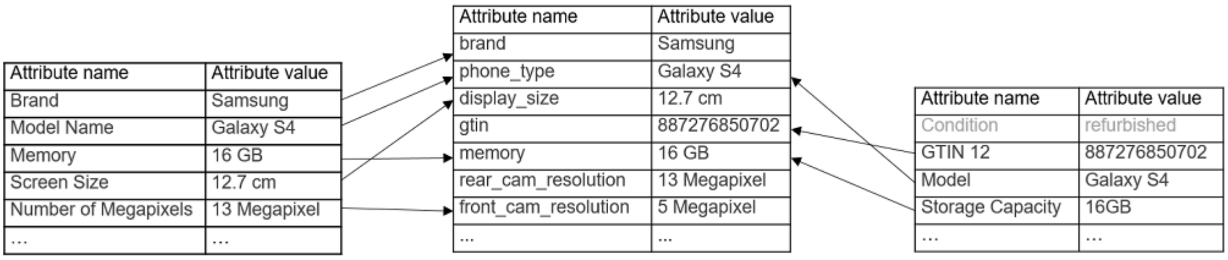


Fig. 1. Examples of product specifications' mappings: (left) Specification from walmart.com, (center) Centralised product Catalog and (right) Specification from ebay.com

As all three algorithms that are introduced in this paper build on GenLink, this section summarises the main components of the GenLink algorithm. The full details of the algorithm are presented in [19].

3.1. Linkage Rule Format

Within GenLink, linkage rules are represented as a tree built out of four basic types of operators: (i) property operators, (ii) transformation operators, (iii) comparison operators and (iv) aggregation operators. The linkage rule tree is strongly typed i.e. only specific combinations of the four basic operators are allowed. Figure 2 shows two examples of linkage rules for matching data describing mobile phones. The formal grammar of the linkage rule format is found in [19].

Property operators. Retrieves all values of a specific property p of each entity. For instance, in Figure 2a the left most leaf in the tree retrieves the value for the *"phone_type"* property from the source dataset.

Transformation operators. Transforms the values of a set of properties or transformation operators. Examples of common transformation functions include case normalization, tokenization, and concatenation of values from multiple operators.

Comparison operators. GenLink offers three types of comparison operators. The first type of operators are character-based comparisons: equality, Levenshtein distance, and Jaro-Winkler distance. The second type includes token-based comparators: Jaccard similarity and Soft Jaccard similarity. The comparison is done over a single property or a specific combination of properties. The third type of comparison operators, numeric-similarity, calculate the similarity of two numbers. Examples of comparison operators can be seen in Figure 2a as the parents of the leaf nodes.

Aggregation operators. Aggregation operators combine the similarity scores from multiple comparison operators into a single similarity value. GenLink implements three aggregation operators. The maximum aggregation operator aggregates similarity scores by choosing the maximum score. The minimum aggregation operator chooses the minimum from the similarity score. Finally, the average aggregation operator combines similarity scores by calculating their weighted average.

Note that these aggregation functions can be nested, meaning that non-linear hierarchies can be learned. For instance, in Figure 2a, four different properties are being compared (*"phone_type"*, *"brand"*, *"memory"* and *"display_size"*). Subsequently, two average aggregations are applied to aggregate scores from phone_type and brand, and memory and display_size, respectively. Finally, a third average aggregation is applied to aggregate scores from the previous aggregators.

Compared to other linkage rule formats, GenLink's rule format is rather expressive, as it is subsuming threshold-based boolean classifiers and linear classifiers, hence allows for representing non-linear rules and may include data transformations which normalize the values prior to comparison [19]. Therefore, it allows rules to closely adjust to the requirements of a specific matching situation by choosing a subset of the properties of the records for the comparison, normalizing the values of these properties using chains of transformation operators, choosing property-specific similarity functions, property-specific similarity thresholds, assigning different weights to different properties, and combining similarity scores using hierarchies of aggregation operators.

3.2. The GenLink Algorithm

The GenLink algorithm starts with an initial population of candidate solutions which is evolved iteratively by applying a set of genetic operators.

Generating initial population. The algorithm finds a set of property pairs which hold similar values before the population is generated. Based on that, random linkage rules are built by selecting property pairs from the set and building a tree by combining random comparisons and aggregations.

Selection. The population of linkage rules is bred and the quality of the linkage rules is assessed by a fitness function relying on user-provided training data. The purpose of the fitness function is to assign a value to each linkage rule which indicates how close the given linkage rule is to the desired solution. The algorithm uses Matthews Correlation Coefficient (MCC) as fitness measure. MCC [26] is defined as the degree of the correlation between the actual and predicted classes or formally:

$$MCC = \frac{tp \times tn - fp \times fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}} \tag{4}$$

The training data consists of a set of positive correspondences (linking entities identifying the same real-world object) and a set of negative correspondences (stating that entities identify different objects). The prediction of the linkage rule is compared with the positive correspondences, counting true positives and false negatives, the negative correspondences, counting false positives and true negatives. In order to prevent linkage rules from growing too large and potentially overfitting to the training data, we penalize linkage rules based on the number of operators:

$$fitness = MCC - 0.05 \times operatorcount \tag{5}$$

Once the fitness is calculated for the entire population, GenLink selects individuals for reproduction by employing the tournament selection method.

Crossover. GenLink applies six types of crossover operators:

1. ***Function crossover.*** The function crossover selects one comparison operator at random in each linkage rule and interchanges the similarity functions between the selected operators.
2. ***Operators crossover.*** The operators crossover is designed to combine aggregation operators from two linkage rules, by selecting an aggregation from each linkage rule and combining their respective comparisons. The crossover selects all comparisons from both aggregations and removes each comparison with a probability of 50%.
3. ***Aggregator crossover.*** In order to learn aggregation hierarchies, the aggregation crossover operator selects a random aggregation or comparison operator in the first linkage rule and replaces it with a random aggregation or comparison operator from the second linkage rule.
4. ***Transformation crossover.*** This crossover builds chains of transformations. To recombine the transformations of two linkage rules the transformation operators of both rules are combined by randomly selecting an upper and a lower transformation operator, recombining their paths via a two point crossover and removing duplicated transformations.
5. ***Threshold crossover and Weight crossover.*** The last two types of crossovers are used to recombine thresholds and weights respectively, for a random comparison operator in each linkage rules, by averaging their thresholds/weights.

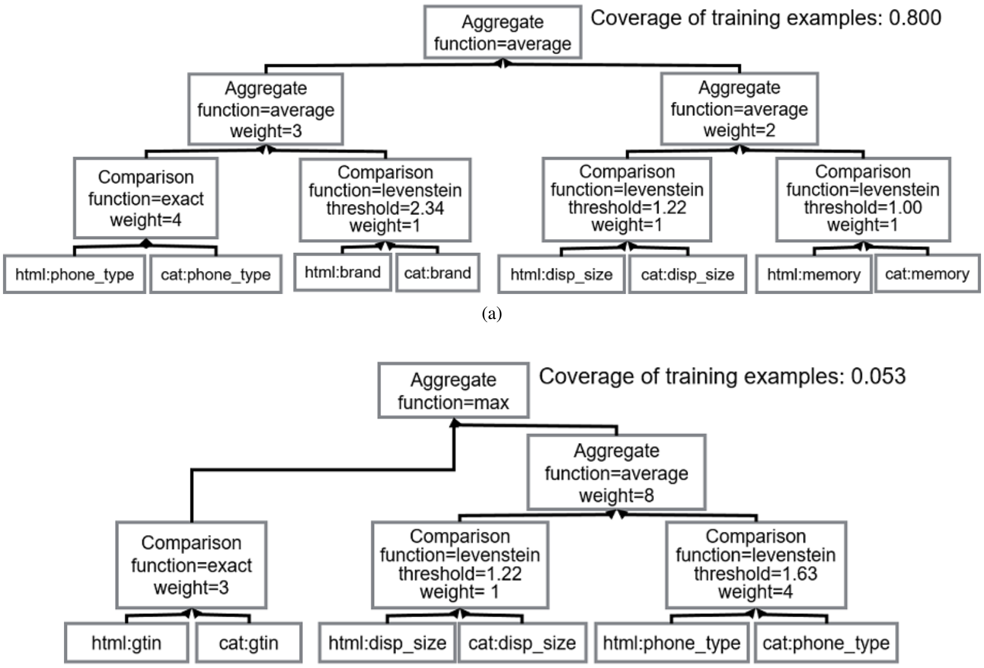
An in-depth discussion of the crossover operators is provided in [19].

4. Approaches

In [37] we have shown that the GenLink algorithm struggles to optimise property selection for sparse datasets. On an e-commerce dataset containing many low-density attributes the algorithm only reached a F-measure of less than 80%, in contrast to the above 95% results that are often reached on dense datasets. In the following, we propose three algorithms that build on the GenLink algorithm and enable it to properly exploit sparse properties. The *GenLinkGL* algorithm builds a group of matching rules for the given matching task (*group generation*) and applies the group of matching rules to create new correspondences (*group application*). Next we introduce *selective aggregations*, new operators within the GenLink algorithm that can better deal with missing values. Finally, we introduce *GenLinkComb* approach, integrating the central ideas of the previous two methods into a single combined method.

4.1. The GenLinkGL Algorithm

The GenLink algorithm lacks the capability to optimise property selection when dealing with sparse data. The algorithm will select a combination of dense properties while sparse properties will rarely be selected.



(a)

(b)

Fig. 2. Example of two rules from the group for the phone category together with the coverage of each rule

This behavior influences adversely cases in which values from relatively dense properties are missing. For instance, when matching product data describing mobile phones from different e-shops, the brand, phone type, and memory properties will be rather important for the matching decisions and these attributes will also likely be rather dense as they are provided by many e-shops. Therefore, GenLink will focus on these attributes and due to the penalty on large rules (compare Equation 5) will not include alternative attribute combinations involving low-density properties, such as gtin[1], display size, or operating system. In cases in which a value of one of these dense attributes is missing, the algorithm will likely fail to discover the correct match, while by exploiting a combination of alternative low-density attributes it would have been possible to recognize that the both records describe the same product. Including all alternative attribute combinations into a single linkage rule would result in rather large rules containing multiple alternative branches that encode the different attribute combinations. Due to the penalty for large rules from Equation 5, only the most important alternative attribute combinations will be included into the rules, whereas combinations having a lower coverage will be left unused.

A way to deal with this problem could be to loosen the size penalty in Equation 5, however removing (or loosening) the penalty has the potential to result with an overfitted model and thus would not improve results. With *GenLink Group Learning* (GenLinkGL), we choose an alternative approach - instead of trying to grow very large rules that cover different attribute combinations, we learn sets of rules in which each rule is optimized for a specific property combination. The method allows us to separate more clearly the issue of avoiding overfitting rules while still being able to cover multiple property combinations. By combining multiple combinations of properties in a group, the learning algorithm is given the freedom to optimize matching rules not only for the most common attribute combinations, but also for less common combinations involving sparse properties, as a result increasing the overall recall. In the following, we describe how GenLinkGL combines rules into groups and later selects a rule from the group in order to match a pair of records having a specific property combination.

Group generation. The basic idea of the first algorithm, presented in Algorithm 1, is that by grouping different linkage rules with different properties

[1]Global Trade Item Number (GTIN) is an identifier for trade items, developed by GS1. – www.gtin.info/

Algorithm 1 Generating a group

Input:
$Group \leftarrow rule$ top fitness matching rule
$P \leftarrow Rules$ All matching rules in the available population
Output:
The fittest group
for all $i \in P$ **do**
  if $i.properties \not\subset Group.properties$ **then**
    $PotentialGroup \leftarrow insert(Group, i)$
    if $fitness(PotentialGroup) > fitness(Group)$ **then**
      $Group = PotentialGroup$
    end if
  end if
end for
return G

we could circumvent the missing values in the data. The initial group is populated with the fittest individual from the population generated by GenLink. Subsequently, an initial fitness for this group is computed using the MCC (compare Equation 4).

Motivated by the GenLink algorithm, our algorithm builds a group that maximises fitness. To do that at each learning iteration, the algorithm iterates through the entire population of linkage rules and combines their individual fitness. We restrict the combination to linkage rules whose properties are not a subset of the properties of the group and include a linkage rule that has at least one new property that is not present in the group. We combine the fitness of the linkage rules by summing the number of correctly predicted instances in the training set (compare Equations 6 and 7), calculating for each individual the percentage of the coverage of training examples in the group. Once the correctly predicted instances are summed the current fitness function is applied to the group. If the fitness of that combination is greater than the current fittest group, the new group becomes the best group. As an output the algorithm gives the fittest group.

$$\begin{aligned} tp_{group} &= \sum_{i=1}^{i=|G|} distinct\ tp_i, \\ fp_{group} &= |R_+| - tp_{group} \end{aligned} \tag{6}$$

$$\begin{aligned} tn_{group} &= \sum_{i=1}^{i=|G|} distinct\ tn_i, \\ fn_{group} &= |R_-| - tn_{group} \end{aligned} \tag{7}$$

Algorithm 1 can potentially lead to groups containing a large number of rules, up to the complete population of learned rules. In such case the algorithm is prone to overfitting, since the population might capture the entire training set. In order to prevent this, we penalize groups containing a large number of rules: $fitness_{group} = MCC_{group} - c * rulecount$. Where, $c = (0.001, 0.003, 0.005)$ is a small constant, strictly depending on the number of individuals in the population. The larger the population, the bigger the chance for overfitting. Therefore, the constant should be higher for larger populations in order to penalise the fitness more. By penalizing the fitness by the number of members in the group we ensure that there will be no unneeded bloating of the learned group.

For example, let the linkage rule in Figure 2a be the fittest individual after the $n - th$ learning iteration of the algorithm. The initial group contains this linkage rule. The group would not be able to correctly predict correspondences that could only have been matched by a combination of the gtin, phone_type and memory properties. At the first iteration we combine the group with the linkage rule in Figure 2b containing the gtin property. As a result, the correspondences above could be captured by the group leading to better fitness.

Group application. As an input the second algorithm, presented in Algorithm 2, takes the output of Algorithm 1 and a set of pairs to be matched. The individuals in the input group are sorted by the percentage of coverage. Sorting enables Algorithm 2 to find the more influential individual rules in less iterations. For each pair the algorithm iterates through the group of matching rules. If the pair to be matched contains the same properties as in the matching rule, the matching rule is applied. If there is no matching rule which has the exact properties as the instances, the top matching rule is applied. For instance, when matching *(a) the specification from walmart.com with the product catalog* and *(b) the specification from ebay.com with the product catalog* from Figure 1, the algorithm would use the first rule from Figure 2 for the a pair, but use the second matching rule from Figure 2 for the b pair since in b one of the specifications does not have a value for

Algorithm 2 Applying a group to set of pairs for matching

Input:
$G \leftarrow group$ of matching rules
$Pairs \leftarrow$ pairs for matching
Output:
Linked instances
$Result \leftarrow nil$
for all $pair \in Pairs$ **do**
  for all $rule \in G$ **do**
    if $pair.properties \equiv rule.properties$ **then**
      $Result \leftarrow match(pair, rule)$
      break
    end if
  end for
  if $\nexists match$ **then**
    $Result \leftarrow match(pair, G.top)$
  end if
end for
return $Result$

the display_size attribute, however it contains a gtin attribute.

Property diversity is an underlying factor behind this method. Since the prime goal is to enlarge the combination of properties that are used for matching, it is imperative that the dataset contains a diverse range of properties. More precisely, if the dataset has a smaller number of properties, the number of combination of properties that can be made by grouping linkage rules is smaller. Therefore, this approach would not improve much upon GenLink when dealing with datasets with smaller number of properties.

4.2. The GenLinkSA Algorithm

An alternative to learning groups of small rules specializing on a specific property combination each is to learn larger rules covering more properties and apply a penalty for the uncertainty that arises from values missing in these properties. For instance, a larger rule could rely on five properties for deciding whether two records match. If two of the five properties have missing values, the remaining three properties can still be used for the matching decision. Nevertheless, a decision based on three properties should be considered less certain than a decision based on five properties. In order to compensate for this uncertainty, we could require the values of the remaining three properties to be more similar than the values of the five properties in the original case in order to decide for a match. The *GenLink Selective Aggregations* (GenLinkSA) algorithm implements this idea by changing the behavior of the comparison operators as well as the aggregation operators in the original GenLink algorithm.

Null-enabled Comparison Operators. The original GenLink algorithm does not distinguish between a pair of different values and a pair of values containing a missing value. In both cases, the algorithm assigns the similarity score 0. This is problematic when similarity scores from multiple comparison operators are combined using the aggregation function average or minimum, as the resulting similarity score will be unnaturally low for the case of missing values. In order to deal with this problem, GenLinkSA amends the comparison operators with the possibility to return the value *null*: a GenLinkSA comparison operator will return *null* if one or both values are missing. If both values are filled, the operator will apply its normal similarity function and return a value in the range $[0, .., 1]$.

Selective Aggregation Operators. The GenLink aggregation operators calculate a single similarity score from the similarity values of multiple comparison operators using a specific aggregation function such as weighted average, minimum, or maximum. GenLinkSA adjusts the aggregation operators to apply the aggregation function only to non-null values. In order to compensate the uncertainty that results from missing values (comparison operators returning the value *null*), the similarity score that results from the aggregation is reduced by constant factor α for each comparison operators that returns a *null* value. In this way, all non-null similarity scores are aggregated and a penalty is applied for each property pair containing missing values. Formally, a GenLink aggregation is defined by the following:

$$\begin{aligned} & S^a : (S^* \times N^* \times F^a) \rightarrow S \\ & (\bar{s}, \bar{w}, f^a) \rightarrow ((e_a, e_b) \rightarrow f^a(s_e, w)) \\ with \quad & s_e : (s_1(e_a, e_b), s_2(e_a, e_b), .., s_n(e_a, e_b)) \end{aligned} \tag{8}$$

The first argument S^* contains the similarity scores returned by the operators of this aggregation while the second argument N^* contains a weight for each of the operators, finally the third argument F^a represents the aggregation function that is applied to compute the similarity score S.

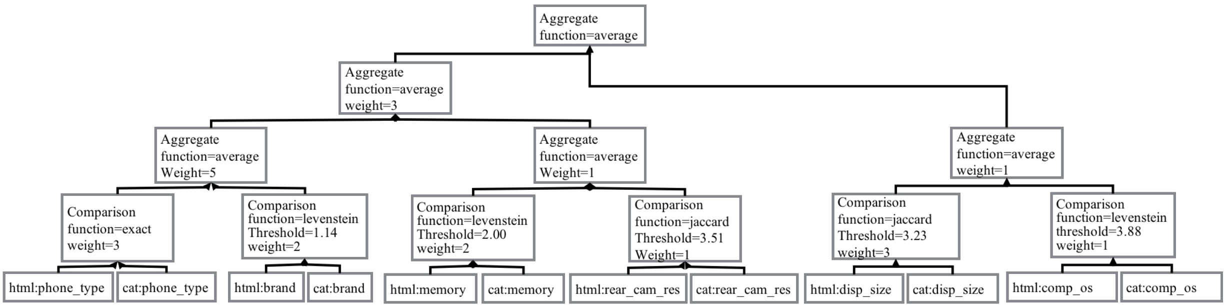


Fig. 3. GenLink SA learned rule for the Phone category

Given the aggregation operators, we can now define GenLinkSA's *selective aggregation operators* as:

$$\begin{aligned} S^a &: (S^* \times N^* \times F^a) \rightarrow S \\ (\bar{s}, \bar{w}, f^a) &\rightarrow ((e_a, e_b) \rightarrow f^a(s_e, w)) - \upsilon \\ \textit{with} \quad s_e &: (s_1(e_a, e_b), s_2(e_a, e_b), .., s_n(e_a, e_b)), \\ \upsilon = \beta \times &\mid \{\, s_i(e_a, e_b) \mid s_i(e_a, e_b) \rightarrow \textit{null} \wedge s_i \in s_e \,\} \mid \end{aligned} \tag{9}$$

Where the uncertainty factor υ is defined as the number of *null* values multiplied by a small valued constant factor $\beta = (0.01, 0.03, 0.05)$. The uncertainty factor serves to penalize the rule for each null similarity operator. As the overall similarity score is reduced by the uncertainty factor, the values of the non-null properties must be more similar in order to reach the same similarity score as for a pair in which all properties are filled.

For example, let the rule learned by the GenLinkSA algorithm be the one shown in Figure 3 and let instances for matching be *(a) the specification from walmart.com that should be matched with the product catalog* and *(b) the specification from ebay.com to be matched with the product catalog* from Figure 1. When matching (a) only a small penalty will be applied since for five out of six comparisons a non-null similarity score will be returned and only the comparison for one property (*comp_os*) will be penalised. On the other hand, the pair (b) will be heavily penalised since four of the six comparisons will return null values. Evidently, this method will discourage high similarity scores in the presence of missing values and will thus refrain from considering borderline cases with missing values as matches, resulting in a higher precision.

4.3. The GenLinkComb Algorithm

GenLinkGL and *GenLinkSA* tackle the issue of missing values differently. Namely, GenLinkGL strives to group matching rules exploiting different combinations of properties and thus is able to apply alternative rules given that values of important properties are missing. By being able to exploit alternative property combinations, GenLinkGL is tailored to improve recall. On the other hand, by penalizing comparisons with missing values, GenLinkSA incentivises learning matching rules that include more properties and substantially lowers the similarity scores of uncertain pairs, and by that improves precision. As the basic ideas behind GenLinkGL and GenLinkSA do not exclude each other but are complementary, a combination of both methods into a single integrated method could combine their advantages: optimize rules for alternative attribute combinations while at the same time dealing with the uncertainty that arises from missing values inside the rules. The *GenLinkComb* algorithm achieves this by combining the GenLinkSA and the GenLinkGL algorithms as follows: GenLinkComb uses the GenLinkSA algorithm to evolve the population of linkage rules. In each iteration of the learning process, GenLinkComb groups the learned rules together using the GenLinkGL algorithm. By being able to deal with missing values either inside the rules using the selective aggregation operators or within the grouping of rules, the GenLinkComb learning algorithm has a higher degree of freedom in searching for a good solution.

5. Evaluation

The evaluation of the aforementioned methods was conducted using six benchmark datasets: three e-commerce product datasets, and three other datasets describing restaurants, movies, and drugs. In addition to comparing GenLinkGL, GenLinkSA, and GenLinkComb with each other, we also compare the approaches to existing systems including CoSum-P, FEBRL, EAGLE, COSY, MARLIN, ObjectCoref, and RiMOM. The following section will describe the six benchmark datasets, give details about the experimental setup, and present and discuss the results of the matching experiments.

5.1. Datasets

Product Matching Datasets. We use three different product datasets for the evaluation:

Abt-Buy dataset: The dataset includes correspondences between 1081 products from Abt.com and 1092 from Buy.com. The full input mapping contains 1.2 million correspondences, from which 1000 are annotated as positive correspondences (matches). Each entity of the dataset might contain up to four properties: product name, description, manufacturer and price. The dataset was introduced in [22]. Since the content of the product name property is a short text listing various product features rather than the actual name of the product, we extract the product properties shown in Table 1 from the product name values using the dictionary-based method presented in [39]. We choose the Abt-Buy dataset because it is widely used to evaluate different matching systems [5, 9].

Amazon-Google dataset: The dataset includes correspondences between 1363 products from Amazon and 103,226 from Google. The full input mapping contains 4.4 million correspondences, from which 1000 are annotated as matches. Each entity of the dataset contains the same properties as the Abt-Buy dataset. This dataset is presented in [22]. We perform the same extraction of properties as in the Abt-Buy dataset. The Amazon-Google data set has also been widely used as benchmark data set [22].

WDC Product Matching Gold Standard: This gold standard [38] for product matching contains correspondences between 1500 products (500 each from the categories headphones, mobile phones, and TVs), collected from 32 different websites and a unified product catalog containing 150 products with the following distribution: (1) Headphones - 50, (2) Phones - 50, and (3) TVs - 50. The data in the catalog has been scraped from leading shopping services, like Google Shopping, or directly from the vendor's website. The gold standard contains 500 positive correspondences (matches) and more than 25000 negative correspondences (non-matches) per category. Compared to the Amazon-Google and Abt-Buy datasets, the WDC Product Matching Gold Standard is more heterogeneous as the data has been collected from different websites. The gold standard also features a richer integrated schema containing over 30 different properties for each product category.

Table 1

Properties together with their density in the Abt-Buy and Amazon-Google datasets.

Dataset	Property	Density (A / B) %
Abt-Buy	Original Attributes	
	Product Name	100
	Description	63
	Manufacturer	48
	Price	36
	Extracted Attributes	
	Model	91
	Brand	72
Amazon-Google	Original Attributes	
	Product Name	100
	Description	70
	Manufacturer	52
	Price	31
	Extracted Attributes	
	Model	88
	Brand	76

Other Entity Resolution Datasets. In order to be able to compare our approaches to more reference systems, as well as to showcase the ability of our algorithms to perform on datasets from different application domains, we run experiments with three additional benchmark datasets which were used in [19]:

Restaurant dataset: The dataset contains correspondences between 864 restaurant entities from the Fodor's and Zagat's restaurant guides. Specifically, 112 duplicate records were identified.

Sider-Drugbank dataset: The dataset contains correspondences between 924 drug entities in the Sider dataset and 4772 drug entities in the Drugbank dataset. Specifically, there have been 859 duplicate records identified.

LinkedMDB dataset This dataset contains 100 correspondences between 373 movies. The authors note that special care was taken to include relevant corner cases such as movies which share the same title but have been produced in different years.

Table 2

Properties and property density of the WDC Product Matching Gold Standard, Restaurants, Sider-Drugbank and LinkedMDB datasets. Note that properties having a density below 10% are not included into the table.

Dataset	Property	Density (A / B) %
WDCPr Gold Standard	Headphones	
	Brand	97 / 100
	Item Type	87 / 100
	MPN	60 / 86
	Color	56 / 96
	Sensitivity	53 / 88
	Impedance	53 / 92
	Cup Type	47 / 38
	Form Factor	43 / 77
	Magnet Mat.	27 / 51
	Diaphragm	25 / 35
	Phones	
	Phone Type	91 / 100
	Memory	87 / 95
	Brand	86 / 100
	Color	79 / 43
	Display Size	71 / 92
	Rear Cam. Res.	70 / 85
	OS	64 / 64
	Display Res.	48 / 53
	Processor	28 / 36
	Front Cam. Res.	20 / 66
	TVs	
	Brand	100 / 100
	Item Type	91 / 100
	Display Type	81 / 85
	Display Size	65 / 96
	Display Res	55 / 87
	Tot. Size	51 / 74
	Ref. Rate	50 / 96
	Img. Asp. Rat.	38 / 60
	Connectivity	35 / 61
	Resp. Time	10 / 25
Restaurant	Name	100
	Address	100
	Contact	100
	Type	100
Sider-Drugbank	Name	100 / 100
	Indication	100 / 93
LinkedMDB	Name	100 / 100
	Director	100 / 100
	Rel Date	100 / 100
	Studio	95 / 97

Tables 1 and 2 give an overview of densities of properties in the six evaluation datasets. If the density of a property differs in the source (A) and the target (B) dataset, both densities are reported. For the Abt-Buy and Amazon-Google datasets, we show all original property densities as well as the density of the extracted properties. As stated before, the product datasets exhibit more sparsity. The Abt-Buy and Amazon-Google datasets follow a similar distribution in which only the product name property has a density of 100%. It is worth to note that the product name property in these datasets is actually a short description of the product mentioning different properties rather than the actual product name. WDC Product Matching Gold Standard contains a small set of properties with a density above 90% while most properties belong to the long tail of rather sparse properties [38].

5.2. Experimental Setup

Baselines. As baselines for the WDC dataset, we repeat TF-IDF cosine similarity and Paragrph2Vec experiments presented in [38], additionally we learn a decision tree and a random forest as baselines. The first baseline, considers pair-wise matching of product descriptions for which TF-IDF vectors are calculated using the bag-of-word feature extraction method. The second baseline, considers building a Paragraph2Vec model [24] for product names using 50 latent features and the Distributed Bag-of-Words model. Decision trees and random forests are learned in Rapidminer[2] using grid search parameter optimization as well as offering the learning algorithm different similarity metrics (e.g. Jaro-Winkler, Jaccard, numeric).

Other Entity Resolution Systems. In order to set the GenLink results into context, we also ran the WDC Gold Standard experiments with EAGLE [34], a supervised matching system that also employs genetic programming[3], FEBRL [9][4], an entity resolution system that internally employs an SVM, and CoSum-P [44], an unsupervised system that treats entity resolution as a graph summarization problem. We pre-compute attribute similarities for CoSum-P as described in [44].

Additionally, we provide a comparison to handwritten Silk rules. These rules are composed of up to six properties for each product category and were written by the authors of the article using their knowledge about the respective domains as well as statistics about the datasets. As an example, the handwritten rule that was used for matching headphones implements the idea that if the very sparse properties html:gtin or html:mpn match exactly, the record pair should be considered as a match. If the numbers are not present or do not match, the rule should fall back to averaging the similarity of the properties html:model,

[2]RapidMiner is a data science software platform - https://rapidminer.com/
[3]http://aksw.org/Projects/LIMES.html
[4]https://sourceforge.net/projects/febrl/

html:impedence and html:headphone_cup_type giving most weight to html:model.

GenLinkGL, GenLinkSA, and GenLinkComb. The GenLinkGL, GenlinkSA, and GenLinkComb algorithms were implemented on top of the Silk Framework[5]. The source code of the original GenLink implementation[6] as well as the source code of GenLinkGL, GenlinkSA, GenLinkComb algorithms[7] is publicly available, so all results presented in this article can be replicated. Table 3 gives an overview of the aggregation, comparison, and transformation functions the algorithms could choose from in the experiments. It should be noted that for each aggregation operator there exists also a selective aggregation operator. Hyper parameters are set using grid search. Even though grid search was run for each dataset, the resulting parameter values were the same for all datasets. Table 4 summarises the parameters that were used for GenLink and its variants in the experiments. All experiments are run 10 times and the results are averaged.

GenLink and its variants as well as EAGLE were trained on a balanced dataset consisting of 66% positive correspondences and the same number negative correspondences. The systems were evaluated afterwards using the remaining 33% of the correspondences. For training FEBRL, we calculated TF-IDF scores and cosine similarity for all pairs given in the dataset. As with GenLink and EAGLE, FEBRL was trained on 66% of the data and evaluated on the rest. For the experiments on the Abt-Buy and Amazon-Google datasets, all systems were trained using the original as well as the extracted attribute-value pairs.

All experiments have been conducted using a Intel(R) Xeon CPU with 6 cores available while the Java heap space has been restricted to 4GB. On average, it took GenLink and GenLinkSA approx. 6 minutes to learn a matching rule using the maximum number of iterations (see Table 4), while it took GenLinkGL and GenLinkComb 8.5 minutes to learn a matching rule.

Preprocessing. The restaurants, movies, and drugs datasets have an original density of over 90%. In order to use them to evaluate how the different approaches perform on sparse data, we systematically removed 25%, 50% and 75% of the values. More precisely, we first randomly sample 50% of properties (not including the *name* property) and for those we randomly select 25%, 50% and 75% of the values and removed the rest, thus introducing greater percentage of null values in the datasets. We do not remove values from all properties since we want to recreate the sparseness as in the product datasets as close as possible. We do not remove the name property since it is the only relevant identifier for a human, i.e without it even a human cannot decide whether two entities are the same.

[5]www.silkframework.org

[6]https://github.com/silk-framework/silk. To be noted that the 2.6.0 version was used for the experiments.

[7]https://github.com/petrovskip/silk.2.6-GenLinkSA and https://github.com/petrovskip/silk.2.6-GenLinkGL

Table 3

Available aggregation comparison and transformation functions. The transformation functions are used only for non-product datasets

Comparison	Aggregation	Transformations
Exact Similarity	Average	Tokenize
Levenstein Distance	Maximum	Lower Case
Jaccard Similarity	Minimum	Concatenate
Number Similarity		

Table 4

GenLink (GL/SA/Comb) Parameters

Parameter	Value
Population size	1000
Maximum iterations	100
Selection method	Tournament selection
Tournament size	10
Probability of Crossover	50%
Probability of Mutation	50%
Stop Condition	F-measure = 1.0
Matching Rule Penalty	0.03
Uncertainty constant	0.05

5.3. Product Matching Results

Table 5 gives an overview of the matching results on the WDC Product Matching Gold Standard dataset. As baselines, we take TF-IDF cosine similarity and Paragrph2Vec experiments presented in [38], and decision tree and random forest explained above. Moreover, we compare results from: (i) handwritten matching rules, (ii) the GenLink algorithm, (iii) GenLinkGL, (iv) GenLinkSA and (v) GenLinkComb. Additionally, we compare to three state-of-the-art matching systems for this dataset: (i) EAGLE [34], (ii) FEBRL [22] and (iii) CoSum-P [44] as explained above.

As expected both baselines perform poorly for each product category. Specifically, TF-IDF could not capture enough details of a given entity. Paragaph2Vec, improves on the TF-IDF baseline by including the semantic relations between the words of a given record. However, the semantic relationships do not prove to be sufficient. The third baseline however, a decision tree approach, is already an adequate baseline as it consistently comes close to the handwritten rules. Moreover, the random forest model gives very good results on all

Table 5

Matching results per category for the WDC Product Matching Gold Standard

Headphones			
	Precision	Recall	F-measure
Baseline TF-IDF Cosine	0.622	0.559	0.588
Baseline Pargraph2vec	0.667	0.685	0.675
Baseline Decision Tree	0.892	0.712	0.791
Baseline Random Forest	0.891	0.764	0.822
Handwritten Rule	0.841	0.838	0.839
EAGLE [34]	0.661	0.905	0.763
GenLink [19]	0.692	0.946	0.799
CoSum-P [44]	0.795	0.868	0.830
FEBRL [9]	0.884	0.837	0.850
GenLinkGL	0.837	0.924	0.888
GenLinkSA	**0.922**	0.925	0.923
GenLinkComb	0.920	**0.961**	**0.940**
Phones			
	Precision	Recall	F-measure
Baseline TF-IDF Cosine	0.385	0.676	0.491
Baseline Pargraph2vec	0.497	0.624	0.553
Baseline Decision Tree	0.751	0.600	0.667
Baseline Random Forest	0.771	0.726	0.747
Handwritten Rule	0.656	0.722	0.687
EAGLE [34]	0.699	0.672	0.685
GenLink [19]	0.708	0.715	0.712
CoSum-P [44]	0.746	0.821	0.781
FEBRL [9]	0.792	0.748	0.776
GenLinkGL	0.742	**0.894**	0.808
GenLinkSA	0.813	0.737	0.773
GenLinkComb	**0.815**	0.886	**0.849**
TVs			
	Precision	Recall	F-measure
Baseline TF-IDF Cosine	0.661	0.474	0.554
Baseline Pargraph2vec	0.654	0.553	0.572
Baseline Decision Tree	0.839	0.714	0.771
Baseline Random Forest	0.785	0.810	0.797
Handwritten Rule	0.782	0.716	0.747
EAGLE [34]	0.722	0.674	0.697
GenLink [19]	0.790	0.711	0.748
CoSum-P [44]	0.779	0.814	0.796
FEBRL [9]	0.807	0.747	0.775
GenLinkGL	0.791	**0.875**	0.819
GenLinkSA	**0.864**	0.745	0.810
GenLinkComb	0.863	0.815	**0.838**

three datasets. With that said, it is to be expected to have better results for both decision tree and random forest with a better feature extraction model as proven in Ristoski et al. [40].

EAGLE [34] and GenLink [19] improve on the baselines since they have the ability to optimise the thresholds for comparisons and the weights within aggregations. Both methods have comparable results with the handwritten rules. The first method that shows a better performance than the handwritten rules for all product categories is FEBRL [9]. Because of FEBRL's SVM implementation is optimized for entity resolution, the system seems to be able to capture more nuanced relationships between data points than the handwritten rules. The main difficulty of the FEBRL is recall. In addition, the method has problems with matching corner cases.

The more recent approach, CoSum-P [44], overcomes the results of FEBRL. The graph summarization approach is able to successfully generalise entities based on pair-wise pre-computed property similarities that refer to the same entity into one super node. However, having no supervision (ability to learn from negative examples) the algorithm suffers from lower precision due to the inability to distinguish between closely related entities. For instance, "name: iphone 6; memory: 16gb" and "iphone 6s; memory: 16gb" would give a high pre-computed similarity score, and thus will be clustered together. Without negative references there is no way for the approach to differentiate between these two products.

All of the GenLinkGL, GenLinkSA, and GenLinkComb consistently outperform results to CoSum-P, FEBRL and the handwritten rules, according to the Friedman [non-parametric rank] test [15] with significance level of $0.01 \leqslant p \leqslant 0.05$. Additionally, they consistently show significant improvement over EAGLE and GenLink according to the McNemar's test [29] with significance level of $p \leqslant 0,01$. For instance, when comparing FEBRL to the GenLinkGL algorithm, we can notice significantly worse recall results. The GenLinkGL algorithm decreases the number of false negatives by learning sets of rules in which each rule is optimized for a specific property combination. Hence, the algorithm is successfully circumventing missing values, and in turn exhibits a jump in recall. Correspondingly, the GenLinkSA algorithm gives comparable results in F-measure compared to FEBRL, mostly due to the jump in precision. The precision jump occurs since the selective aggregation operators substantially lower matching scores of uncertain pairings due to the uncertainty factor. Due to this penalty, pairs with missing values which otherwise would have borderline similarity will not be considered matches. Both the jump in recall of GenLinkGL and the jump in precision of GenLinkSA contribute to improve the matching and the algorithms have comparable results in F-measure. Finally, GenLinkComb shows significantly better performance in F-measure than the rest of the tested field, due to the fact that the combination method is able of both preserving precision by penalising borderline cases with missing values and

Table 6
Standard deviation of the GenLink Algorithms on the WDC dataset

Headphones		
	Average F-score	Standard Dev.
GenLink	0.799	±0.054
GenLinkGL	0.888	±0.029
GenLinkSA	0.923	±0.051
GenLinkComb	0.940	±0.034
Phones		
	Average F-score	Standard Dev.
GenLink	0.712	±0.092
GenLinkGL	0.804	±0.035
GenLinkSA	0.773	±0.095
GenLinkComb	0.849	±0.039
TVs		
	Average F-score	Standard Dev.
GenLink	0.748	±0.087
GenLinkGL	0.819	±0.042
GenLinkSA	0.910	±0.087
GenLinkComb	0.838	±0.047

preserving recall by successfully exploiting alternative attribute combinations.

Category wise, the headphones category proves to be an easier matching task obtaining the best results with 94% F-measure. Headphones have a smaller number of distinct properties and therefore e-shops tend to more consistently describe products with the same attributes compared to the other two categories. The TVs and phones category reach similar F-measures of 83.8% and 84.9% respectively.

Table 6 shows the averaged results of the algorithms and their standard deviation values. The stability of GenLink and GenLinkSA is improved by GenLinkGL and GenLinkComb. The latter, group multiple individuals, thus increasing the probability to converge to the optimal solution.

Comparison of the learned matching rules. In order to explain the differences in the results of GenLinkSA, GenLinkGL, and GenLinkComb, we analyze and compare the rules that were learned by the three algorithm for matching using the expample of mobile phones. Figure 3 shows the GenLinkSA rule that was learned. As we can see, the rules uses six properties which are combined using a hierarchy of average aggregations. Within the hierarchy, more weight is put onto a branch containing four properties, as well as on the properties brand and phone_type within this branch. The GenLinkGL algorithm has learned a group consisting of 12 matching rules that use 15 distinct properties for matching phones. Table 7 shows the top five rules from the GenLinkGL approach sorted by their coverage. More than 50% of the rules contain the model (phone_type) and the display size (disp_size) attributes. It is interesting to examine the coverage of the learned rules: The first rule was applied to match 80% of the pairs in the training data. The second rule was only used for 5% of the cases, the next rule for 2% and so on, meaning that the data contained one dominant attribute combination (the one exploited by the first rule) while by specializing on alternative combinations (like the second rule involving the gtin property) still improved the overall result. Furthermore, most of the learned matching rules use similar combinations of aggregation functions (average aggregation). The only exception is the second rule which uses the property gtin. Namely, the gtin property by itself is enough to identify the specific product, thus the maximum aggregation function is used. For matching phones, the GenLinkComb algorithm has learned a group that only consists of five matching rules which use 10 distinct properties. Consequently it achieves a better F1-performance using less rules and less properties compared to GenLinkGL. Table 8 shows the rules that were learnt by the GenLinkComb algorithm, again sorted by coverage. Interestingly, the rules have a more homogenous coverage distribution than the GenLinkGL rules. Instead of generating low-coverage rules for exotic property combinations as GenLinkGL does, GenLinkComb generate less groups which exploit more properties each and uses the selective aggregations and the uncertainty penalty to deal with missing values within these properties. The property composition also supports this argument: The robust property composition of GenLinkComb suggests that the learned matching rules in the group contain more nuanced differences, while GenLinkGL has more irregular property composition.

Amazon-Google and Abt-Buy Results. To evaluate the algorithms on datasets having lower number of distinct properties (see Table 1), we applied the algorithms to the Amazon-Google and Abt-Buy datasets. The results of these experiments are given in Table 9 and Table 10. As reference systems, apart of FEBRL, the best performing approaches found in literature are listed. Table 9 gives results on the matching experiment done on the Amazon-Google dataset. GenLinkComb outperforms a commercial system [22] based on manually set attribute-level similarity thresholds. The commercial system [22] derives matching rules similar to the handwritten rules in WDC Product Matching Gold Standard and therefore is inferior to the GenLinkComb. CoSum-P [44], shows comparable

Table 7

Property, Comparisons, Aggregations and Training example coverage for the top 5 rules in the learned group for phone category learned by GenLinkGL

Properties	Comps.	1st Agg.	2nd Agg.	Coverage
phone_type	Exact	Avg		
brand	Levens.		Avg	0.800
dips_size	Levens.	Avg		
memory	Levens.			
gtin	Exact		Max	
memory	Levens.	Avg		0.053
phone_type	Levens.			
phone_type	Exact	Avg		
brand	Levens.		Avg	0.020
proc_type	Exact	Avg		
core_count	Exact			
phone_type	Exact	Avg		
comp_os	Levens.		Avg	0.017
rear_cam_res	Jaccard	Avg		
front_cam_res	Jaccard			
disp_size	Exact	Avg		
brand	Exact		Avg	0.013
rear_cam_res	Jaccard	Avg		
disp_res	Jaccard			

Table 8

Property, Comparisons, and Training example coverage and Normalized threshold mean for the top 5 rules in the learned group for phone category learned by GenLinkComb

Properties	Comps.	1st Agg.	2nd Agg.	3rd Agg.	Coverage
phone_type	Levens.	Avg			
brand	Levens.		Min		
memory	Jaccard	Avg		Avg	0.492
dips_size	Jaccard				
memory	Exact.	Min			
phone_type	Levens.				
phone_type	Exact	Min			
memory	Exact.			Min	0.221
rear_cam_res	Jaccard		Avg		
memory	Levens.	Avg			
dips_size	Levens.				
phone_type	Exact	Avg			
brand	Levens.		Avg		
memory	Levens..	Avg		Avg	0.215
rear_cam_res	Jaccard				
dips_size	Jaccard	Avg			
comp_os	Levens.				
phone_tupe	Exact	Min			
memory	Levens.		Min	Avg	
phone_type	Levens.				0.037
proc_type	Exact				
phone_type	Levens.	Min			
memory	Exact				
memory	Levens.	Min		Min	0.035
front_cam_res	Jaccard		Avg		
disp_res	Jaccard	Avg			
phone_type	Jaccard				

results to GenLinkComb. As the datasets only have a low number of properties and as these properties often contain multi-word texts, the token-similarity based approach of CoSum-P can play its strength, leading to much better relative results compared to the WDC Gold Standard (Table 5).

Table 10 gives results on the matching experiment done on the Abt-Buy dataset. As with previous datasets GenLinkComb shows the best performance in terms of F-Measure. Both, FEBRL's SVM classifier [9] and MARLIN [5][8] give comparable results to both GenLinkSA and GenLinkGL. This is to be expected, as the features for both FEBRL and MARLIN were manually engineered for the given datasets whereas our methods select features automatically. Moreover, the SVM's for both FEBRL and MARLIN were trained with larger feature sets than our approaches (five matchers on two properties).

When comparing the results of the experiments with WDC Product Matching Gold Standard to the results of the Abt-Buy and Amazon-Google datasets it becomes evident that the GenLink variants perform better on datasets containing a large number of properties than on dataset containing only a smaller number of properties.

[8]Results from experiments with FEBRL and MARLIN are published in [22]

Table 9

Product matching results for the Amazon-Google dataset

	Precision	Recall	F-measure
GenLink [19]	0.493	0.571	0.513
GenLinkGL	0.501	0.813	0.604
GenLinkSA	0.691	0.632	0.643
GenLinkComb	0.690	0.651	**0.669**
Reference Systems			F-measure
CoSum-P [44]	0.639	0.695	**0.666**
FEBRL [9]			0.601
COSY [22]			0.622

5.4. *Other Domains Results*

Generally, for all datasets we can conclude that our methods find it difficult to find the correct matches when dealing with severely sparse data (25%). Additionally, GenLinkComb and GenLinkSA have similar performance and both tend to outperform GenLinkGL for every dataset for the sparser settings. In contrast, when the datasets have 75% property density, our methods perform close to the results of reference systems achieved on the datasets with more than 90% property density.

Table 10

Product matching results for the Abt-Buy dataset

	Precision	Recall	F-measure
GenLink [19]	0.632	0.694	0.661
GenLinkGL	0.650	0.833	0.730
GenLinkSA	0.721	0.714	0.717
GenLinkComb	0.723	0.798	**0.758**

Reference Systems	F-measure
FEBRL [9]	0.713
MARLIN [5]	0.708

Table 11

Results for the Restaurants dataset

	Density		
	25%	**50%**	**75%**
	F-measure	F-measure	F-measure
GenLink [19]	0.651	0.654	0.909
GenLinkGL	0.642	0.661	0.905
GenLinkSA	0.654	0.660	0.938
GenLinkComb	0.653	0.664	0.936

Reference Systems on original dense dataset	F-measure
GenLink [19]	0.993
Carvalho et al.[7]	0.980

Table 11 gives results on the matching experiment done on the Restaurant dataset. GenLinkSA and GenLinkComb perform closest to the reference systems, while GenLinkGL does not show any improvement on this dataset. Due to low number of properties that this dataset has GenLinkComb and GenLinkGL show little improvement compared to the other methods. Consequently, GenLInkComb and GenLinkGL cannot find enough matching rules with alternative attributes to group, making GenLinkComb to boil down to GenLinkSA and GenLinkGL to boil down to GenLink. Density wise, all three methods follow the same downward trend when the dataset is more sparse, keeping the relative improvements of GenLinkSA and GenLinkGL in comparison to GenLink.

Table 12 gives results on the matching experiment done on the Sider-Drugbank dataset. Even though we systematically lowered the quality of the dataset, GenLink still outperforms the state-of-the-art [18, 43] systems for the case of 75% property density. With that said, GenLinkGL and GenLinkSA reach considerably better results in recall and precision respectively. When the data become severely sparse, like in the case of 25% our methods show an increase of 5% in F-measure compared to GenLink. Similarly to the Restaurant dataset the GenLinkComb does not improve over GenLinkSA as again the grouping algorithm could not find any suitable rules with alternative attributes for grouping.

Table 12

Results for the Sider-Drugbank dataset

	Density		
	25%	**50%**	**75%**
	F-measure	F-measure	F-measure
GenLink [19]	0.345	0.388	0.837
GenLinkGL	0.399	0.424	0.875
GenLinkSA	0.401	0.422	0.871
GenLinkComb	0.402	0.422	0.872

Reference Systems on original dense dataset	F-measure
ObjectCoref [18]	0.464
RiMOM[43]	0.504
GenLink [19]	0.970

Table 13

Results for the LinkedMDB dataset

	Density		
	25%	**50%**	**75%**
	F-measure	F-measure	F-measure
GenLink [19]	0.540	0.587	0.873
GenLinkGL	0.550	0.627	0.911
GenLinkSA	0.559	0.624	0.920
GenLinkComb	0.611	0.658	0.952

Reference Systems on original dense dataset	F-measure
EAGLE [34]	0.941
GenLink [19]	0.999

Table 13 gives results on the matching experiment done on the LinkedMDB dataset, which contains more properties compared to the other two datasets. In this case GenLinkComb outperforms other variations of GenLink even when data spareness is severe. Unlike with the Restaurants and Sider-Drugbank datasets GenLinkComb successfully finds rules with alternative attributes to group and thus increasing F-measure by 5% compared to GenLinkSA.

6. Related Work

Entity resolution has been extensively studied under different names such as record linkage [1, 8, 17, 32], reference reconciliation [13], coreference resolution [25, 31]. In the following, we review a set of representative entity resolution approaches; while we refer to tutorials [16] and surveys [6, 10, 42] for more throughout reviews.

Distance-based entity resolution approaches focus on learning a pairwise distance metric between entities and then either set a distance threshold or build a pairwise classifier to determine which entities are merged. Such pairwise classifiers can be categorised into threshold based boolean classifiers and linear classifiers. One of the first generic approaches for entity resolution based on boolean classifiers is presented at [2]. The approach is based on the assumption that the entity resolution process consists of iterative matching and merging which results in a set of merged records that cannot be further matched or merged with each other. The authors also assume that matching and merging can be done if similar values exists, therefore their approach would not be able to match or merge records with missing values.

One of the most popular method to model distance-based entity resolution approaches is with linear classifiers. There are two popular applications of SVMs to entity matching MARLIN (Multiply Adaptive Record Linkage with INduction) [5] and FEBRL (Freely Extensible Biomedical Record Linkage) [9]. While there are numerous studies that propose approaches for handling missing values in SVMs, for instance [36], these optimizations are often expensive and to our knowledge are not used in matching approaches.

An important use cases of entity resolution is matching of product data. Following the same trend from above various studies show optimization approaches of linear classifiers for product resolution. For instance, Kannan et al. [21] learn a logistic regression model on product attributes extracted from a dictionary model. Similarly, in [23] the authors extend the FEBRL approach from [22] with more detailed features. Finally, in [39], the authors compare various classifiers for product resolution (SVMs, Random Forest, Naive Bayes) with features extracted from a dictionary method and multiple Conditional Random Fields (CRFs) models. The authors, extended their work in [40], where they present extraction models with latent continuous features for product matching and classification, proving that more sophisticated feature extraction methods significantly improve traditional machine learning methods for entity resolution.

The entire process of entity resolution can be unsupervised [11, 27, 35, 44] or supervised [31, 32]. To compare learning entity resolution methods, semi-automatic baseline approaches are used. These approaches are based on a definition of effective linking specifications that excel in one-to-one matching tasks including TF-IDF or Paragraph2Vec with cosine similarity or based on other similarity functions as presented in Hassanzadeh et al. [17]. Limes [32] and Silk [19] are examples of supervised entity resolution systems that focus on combining expressive comparisons with good run-time behavior. Both Limes and Silk learn linkage rules employing similar genetic programming approaches, i.e EAGLE [34] and GenLink respectively. In addition to GenLink, Silk provides ActiveGenLink an active learning approach presented in Isele et al. [20]. As shown throughout this paper, both algorithms do not handle missing values well.

Contrary to the above, in Ngomo et al. [33], the authors present RAVEN - an entity resolution approach based on perceptron learning. Namely, RAVEN treats the discovery of link specifications as a classification problem. It discovers link specifications by first finding class and property mappings between knowledge bases automatically, after which it computes linear and boolean classifiers that can be used as link specifications. However, similar to FEBRL the main limitation of RAVEN is that only linear and boolean classifiers can be learned, making optimization for matching spares data expensive.

There is another direction of work that is focused on collective based entity resolution approaches. For instance, Bhattacharya and Getoor [4] proposed a novel relational clustering algorithm that uses both property and relational information between the entities of same type for determining the underlying entities. However, the defined cluster similarity measure depends primarily on property value similarity, thus missing values will have effect on the cluster similarity measure. Another collective entity resolution approach is introduced in [3] where the authors use an extended LDA model to perform entity resolution for authors and publications simultaneously.

In contrast, [28, 41] use probabilistic model for capturing the dependence among multiple matching decisions. Specifically, CRFs have been successfully applied to the entity resolution domain [28] and is one of the most popular approaches in generic entity resolution. On another hand, a well-founded integrated solution to the entity-resolution problem based on Markov Logic is proposed in [41]. However the approach apply the closed-world assumption, i.e.whatever is not observed is assumed to be false in the world.

One of the first works in the Semantic Web on the topic of unsupervised entity resolution is Nikolov et al. [35]. The authors present a genetic algorithm for matching, similar to EAGLE [34] and GenLink [19]. However, instead of providing reference links as basis for calculating fitness, the authors propose a "pseudo F-measure"; an approximation to F-measure based on indicators gathered from the datasets. Specifically, the fitness function proposed by the author assumes datasets not to contain any duplicates. This assumption is violated by many real world datasets. For instance, the WDC dataset contains many offers for the same product all originating from eBay.

CoSum [44] and idMesh [12] are two representative unsupervised graph-based entity resolution approaches. CoSum and idMesh are both treating entity resolution are graph summarisation problem, i.e. generating super-nodes by clustering entities and in the case of CoSum by applying collective matching techniques. Both approaches employ sophisticated generic similarity metrics. Nevertheless, dues to not using negative evidence, they likely run into problems for use cases in which small syntactic differences matter, such as product type Lul5X versus Lul6X. As shown by the good results of CoSum-P [44] on the Amazon-Google dataset (see Section 5.3), unsupervised approaches can excel in use cases that involve rather unstructured, textual data. But due to not using domain-specific evidence, they likely reach lower relative results for use cases that require domain-specific similarity metrics and attribute weights.

7. Conclusion

The article introduces three methods for learning expressive linkage rules from sparse data. The first method learns groups of matching rules which are each specialized on a specific combination of non-NULL properties. Moreover, we introduce new operators to the GenLink algorithm: *selective aggregation operators*. These operators assign lower similarity values to pairings with missing values which in turn boosts precision. Finally, we presented a method that integrates the central ideas of the previous two methods into one combined method. We evaluate the three methods on six different datasets, three of them are of the e-commerce domain (as one of the domains that often involves sparse datasets), and the other three datasets are benchmark datasets that were used in previous work. We show improvements of up to 16% F-measure compared to handwritten rules, on average 12% F-measure improvement compared to the original GenLink algorithm, 15% compared to EAGLE, 8% compared to FEBRL, and 5% compared to CoSum-P. In addition, we show that the method using *group matching rules* improves recall up to 15%, while *selective aggregation operators* mostly improve precision of up to 16%. The combination that encompasses these methods allows for improvement of up to 5% F-measure compared to the GenLinkGL and GenLinkSA themselves.

As a general conclusion, the high gains in F-measure clearly shows that identity resolution systems should take the use case of sparse data into account and not only focus on dense datasets. When benchmarking and comparing systems, it is important to not only use dense evaluation datasets, but also test on dataset with varying attribute density like the WDC Product Matching Gold Standard [38].

References

[1] Arasu, A., Götz, M., Kaushik, R.: On active learning of record matching packages. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. pp. 783–794. SIGMOD '10, ACM, New York, NY, USA (2010)

[2] Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S.E., Widom, J.: Swoosh: a generic approach to entity resolution. The VLDB Journal—The International Journal on Very Large Data Bases 18(1), 255–276 (2009)

[3] Bhattacharya, I., Getoor, L.: A Latent Dirichlet Model for Unsupervised Entity Resolution, pp. 47–58 (2006)

[4] Bhattacharya, I., Getoor, L.: Collective entity resolution in relational data. ACM Trans. Knowl. Discov. Data 1(1) (Mar 2007)

[5] 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. KDD '03, ACM, New York, NY, USA (2003)

[6] Brizan, D.G., Tansel, A.U.: A. survey of entity resolution and record linkage methodologies. Communications of the IIMA 6(3), 5 (2015)

[7] de Carvalho, M.G., Gonçalves, M.A., Laender, A.H.F., da Silva, A.S.: Learning to deduplicate. In: Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries. pp. 41–50. JCDL '06, ACM, New York, NY, USA (2006)

[8] Christen, P.: Automatic record linkage using seeded nearest neighbour and support vector machine classification. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 151–159. KDD '08, ACM, New York, NY, USA (2008)

[9] Christen, P.: Febrl: A freely available record linkage system with a graphical user interface. In: Proceedings of the Second Australasian Workshop on Health Data and Knowledge Management - Volume 80. pp. 17–25. HDKM '08, Australian Computer Society, Inc., Darlinghurst, Australia, Australia (2008)

[10] Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE transactions on knowledge and data engineering 24(9), 1537–1555 (2012)

[11] Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 475–480. KDD '02, ACM, New York, NY, USA (2002)

[12] Cudré-Mauroux, P., Haghani, P., Jost, M., Aberer, K., De Meer, H.: idmesh: Graph-based disambiguation of linked data. In: Proceedings of the 18th International Conference on World Wide Web. pp. 591–600. WWW '09, ACM, New York, NY, USA (2009)

[13] Dong, X., Halevy, A., Madhavan, J.: Reference reconciliation in complex information spaces. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. pp. 85–96. SIGMOD '05, ACM, New York, NY, USA (2005)

[14] Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Association 64(328), 1183–1210 (1969)

[15] Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association 32(200), 675–701 (1937), https://www.tandfonline.com/doi/abs/10.1080/01621459.1937.10503522

[16] Getoor, L., Machanavajjhala, A.: Entity resolution: theory, practice & open challenges. Proceedings of the VLDB Endowment 5(12), 2018–2019 (2012)

[17] Hassanzadeh, O., Pu, K.Q., Yeganeh, S.H., Miller, R.J., Popa, L., Hernández, M.A., Ho, H.: Discovering linkage points over web data. Proc. VLDB Endow. 6(6), 445–456 (Apr 2013), http://dx.doi.org/10.14778/2536336.2536345

[18] Hu, W., Chen, J., Cheng, G., Qu, Y.: Objectcoref & falcon-ao: Results for oaei 2010. In: Proceedings of the 5th International Conference on Ontology Matching - Volume 689. pp. 158–165. OM'10, CEUR-WS.org, Aachen, Germany, Germany (2010)
[19] Isele, R., Bizer, C.: Learning expressive linkage rules using genetic programming. Proceedings of the VLDB Endowment 5(11), 1638–1649 (2012)
[20] Isele, R., Bizer, C.: Active learning of expressive linkage rules using genetic programming. Web Semant. 23, 2–15 (Dec 2013), http://dx.doi.org/10.1016/j.websem.2013.06.001
[21] Kannan, A., Givoni, I.E., Agrawal, R., Fuxman, A.: Matching unstructured product offers to structured product specifications. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 404–412. KDD '11, ACM, New York, NY, USA (2011)
[22] 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)
[23] Köpcke, H., Thor, A., Thomas, S., Rahm, E.: Tailoring entity resolution for matching product offers. In: Proceedings of the 15th International Conference on Extending Database Technology. pp. 545–550. EDBT '12, ACM, New York, NY, USA (2012)
[24] Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: Xing, E.P., Jebara, T. (eds.) Proceedings of the 31st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 32, pp. 1188–1196. PMLR, Bejing, China (22–24 Jun 2014)
[25] Lee, H., Chang, A., Peirsman, Y., Chambers, N., Surdeanu, M., Jurafsky, D.: Deterministic coreference resolution based on entity-centric, precision-ranked rules. Computational Linguistics 39(4), 885–916 (2013)
[26] Matthews, B.W.: Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure 405(2), 442–451 (1975)
[27] McCallum, A., Nigam, K., Ungar, L.H.: Efficient clustering of high-dimensional data sets with application to reference matching. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 169–178. KDD '00, ACM, New York, NY, USA (2000)
[28] McCallum, A., Wellner, B.: Conditional models of identity uncertainty with application to noun coreference. In: NIPS. pp. 905–912 (2004)
[29] McNemar, Q.: Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika 12(2), 153–157 (1947)
[30] Meusel, R., Petrovski, P., Bizer, C.: The WebDataCommons Microdata, RDFa and Microformat Dataset Series, pp. 277–292. Springer International Publishing, Cham (2014)
[31] Ng, V., Cardie, C.: Improving machine learning approaches to coreference resolution. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. pp. 104–111. ACL '02, Association for Computational Linguistics, Stroudsburg, PA, USA (2002)
[32] Ngomo, A.C.N., Auer, S.: Limes: A time-efficient approach for large-scale link discovery on the web of data. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three. pp. 2312–2317. IJCAI'11, AAAI Press (2011)
[33] Ngomo, A.C.N., Lehmann, J., Auer, S., Höffner, K.: Raven - active learning of link specifications. In: Proceedings of the 6th International Conference on Ontology Matching - Volume 814. pp. 25–36. OM'11, CEUR-WS.org, Aachen, Germany, Germany (2011), http://dl.acm.org/citation.cfm?id=2887541.2887544
[34] Ngonga Ngomo, A.C., Lyko, K.: Eagle: Efficient active learning of link specifications using genetic programming. The Semantic Web: Research and Applications pp. 149–163 (2012)
[35] Nikolov, A., d'Aquin, M., Motta, E.: Unsupervised learning of link discovery configuration. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) The Semantic Web: Research and Applications. pp. 119–133. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
[36] Pelckmans, K., Brabanter, J.D., Suykens, J., Moor, B.D.: Handling missing values in support vector machine classifiers. Neural Networks 18(5âĂŞ6), 684 – 692 (2005), {IJCNN} 2005
[37] Petrovski, P., Bryl, V., Bizer, C.: Integrating product data from websites offering microdata markup. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 1299–1304. WWW '14 Companion, ACM, New York, NY, USA (2014)
[38] Petrovski, P., Primpeli, A., Meusel, R., Bizer, C.: The WDC Gold Standards for Product Feature Extraction and Product Matching, pp. 73–86. Springer International Publishing, Cham (2017)
[39] Ristoski, P., Mika, P.: Enriching Product Ads with Metadata from HTML Annotations, pp. 151–167. Springer International Publishing, Cham (2016)
[40] Ristoski, P., Petrovski, P., Mika, P., Paulheim, H.: A machine learning approach for product matching and categorization. Semantic Web Journal (2017)
[41] Singla, P., Domingos, P.: Entity resolution with markov logic. In: Sixth International Conference on Data Mining (ICDM'06). pp. 572–582 (Dec 2006)
[42] Winkler, W.E.: Matching and record linkage. Business survey methods 1, 355–384 (1995)
[43] Winkler, W.E.: Methods for record linkage and bayesian networks. Tech. rep., Technical report, Statistical Research Division, US Census Bureau, Washington, DC (2002)
[44] Zhu, L., Ghasemi-Gol, M., Szekely, P., Galstyan, A., Knoblock, C.A.: Unsupervised Entity Resolution on Multi-type Graphs, pp. 649–667. Springer International Publishing, Cham (2016)