Brand-Included Model Words Duplicate Detection Method

Noah Molenaar (536419)

Erasmus University Rotterdam
Rotterdam, the Netherlands
536419nm@student.eur.nl

Abstract. In the realm of online shopping, the accessibility of products through comparison-platforms has simplified consumer decision-making. This study enhances the Multi-component Similarity Method with Pre-selection (MSMP) for duplicate detection that can be used by such platforms, building on the pre-selection technique using Locality-Sensitive Hashing (LSH). This paper introduces television brand-names as additional model words to find a better performing, scalable solution for product duplicate detection than the one found for the standard MSMP.

1 Introduction

Shopping and comparing products has become much easier for consumers since the introduction of Web shops. Consumers in the Netherlands for example, can nowadays rely on websites such as `www.vergelijk.nl` and `www.kieskeurig.nl` to find their desired products. These websites compare the same products, offered by different online shops, and present an overview of the prices for these shops and the product specifications. This way, consumers can save up a lot of time by not comparing products among different websites.

A way of implementing duplicate detection demanded by those comparison-websites is the Multi-component Similarity Method (MSM), such as used in [11]. Given that this method can have an enormous running time for vast sets of data, [12] first use Locality-Sensitive Hashing (LSH) as pre-selection method to obtain duplicate candidate-pairs. As MSM will then only be performed on the found candidate-pairs, the computation time gets reduced. The application of LSH from [12] is expanded in [7], by having different model words as input and through the addition of data cleaning. This paper, extends the way LSH is used by [12], by adding television brands to the set of model words. Furthermore, data cleaning is used by implementing inconsistencies such as found in [7]. The research question of this paper is: "*How does the inclusion of brand-name model words contribute to the performance of the Multi-component Similarity Method with Pre-selection?*" The R-code - written together with Jens Peek (542716) - used to answer this question can be found on `https://github.com/NoahMolenaar/CSBA_duplicate_detection.git`.

The remainder of this paper is organized as follows: in Section 2, previous studies related to LSH and MSM are summarized. I present an overview of the methods applied in this paper in Section 3, after which the evaluation of these methods is discussed in Section 4. Finally, I conclude this paper in Section 5.

2 Related Work

The framework of this paper is akin to that presented in [12], which builds upon the Multi-component Similarity Method introduced by [11]. The method is extended by incorporating Locality-Sensitive Hashing (LSH) as a pre-selection technique in [12]. LSH has been proven useful for providing nearest neighbors in vast datasets by [10], while [8] had shown earlier that LSH deals with the rapidly growing search time due to high dimensions. [6] and [12] show that input for the LSH are model words; words containing at least two different characters, being alphanumerical, numerical and/or special characters. LSH can be applied using a signature matrix obtained via random permutations and the selection of the index of the first 1 in the column, such as in [5]. After LSH, the Multi-component Similarity Method (MSM) of [11] can be used while aiming for shorter computation times. Final results are evaluated with the F_1-measure such as in [9] and the slightly adapted version of this measure; the F_1^*-measure.

3 Method Overview

As this paper extends the approach of [12] - the MSMP method, but differs from that of [7] - the MSMP+ method, I refer to my approach as the MSMPb method. The ‘b’ represents brand-names which are implemented in the pre-selection part of MSMP, as clarified in the first subsection. Furthermore, Locality-Sensitive Hashing, Multi-component Similarity Method and evaluation methods are elucidated in this section.

3.1 Data Cleaning

In this paper, I intend to find duplicate TV’s in a dataset consisting of 1624 TV’s, coming from four Web shops: Amazon.com [1], BestBuy.com [2], Newegg.com [4] and TheNerds.net [3]. These Web shops have 163, 773, 668 and 20 TV’s, respectively. Out of the 1624 TV’s, 1262 are unique. Each product has a title and product description containing product-specific features. There are in total 351 different features among all products. In the MSMPb method, model words from the product title as well as from the product description are used. As there are inconsistencies in the data, due to a difference in notation between the stores and due to errors, the data needs to be cleaned in order to get better results. As [7] performed data cleaning for the same data set, I similarly normalize ‘-inch’, ‘-inches’, ‘inches’ and double quotation marks ” to ‘inch’ after decapitalizing all words in the title and product description. Besides, ‘hertz’ is normalized to ‘hz’. Finally, spaces in front of ‘inch’ and ‘hz’ are removed such that they could become model words. In this paper, model words from the title are used to find

the nearest neighbors by locality-sensitive hashing (LSH), just like in [11]. The same regular expression as in [12] to find these model words is used:
[a-zA-Z0-9]*(([0-9]+[ˆ0-9,]+)|([ˆ0-9,]+[0-9]+))[a-zA-Z0-9]*
A model word contains at least two out of three different tokens; being alphanumerical, numerical and special characters. To this list of model words, I add all television brands downloaded from [13], which can be found in the Appendix. The list has been decapitalized and only the first word of the brands have been selected. The addition of brands should contribute to a better pre-selection method of MSM that barely affects the running time, since the set of model words is only increased by 12%, while the part in which this is used, LSH, does not take up much running time.

3.2 Locality-Sensitive Hashing

After creating model words, binary vectors are obtained from these model words similarly as done by [12]. The binary vectors are merged into a binary matrix, which is used to create a signature matrix *M*. As I perform 800 permutations, the signature matrix consists of n=800 rows. LSH can be used to find candidate-pairs with a high similarity, see [10]. M is divided into b bands of r rows, such that $n = b * r$. Candidate pairs consist of products that have been hashed to the same bucket. Different values of b and r will result in a different threshold t, where $t \approx \frac{1}{b}^{\frac{1}{r}}$. On the one hand, a lower t will increase the amount of false positives and reduce the amount of false negatives. On the other hand, a higher t will decrease the amount of false positives and increase the amount of false negatives. The right trade-off between b and r thus needs to be found before the candidate-pairs can be passed on to the MSM.

3.3 Multi-component Similarity Method

MSM is a hierarchical adopted single-linkage clustering method consisting of a triadic function that calculates the similarity between candidate-pairs. The triadic function consists of a part that compares matching key-value pairs, a part that uses the HSM method [6] on non-matching key-value pairs and a part that uses the Title Model Words Method. The algorithm used for MSM is presented clearly by [11]. Just like in that paper, I have used tokens of $q = 3$ characters in the q-gram similarity measure in the first part of the triadic function. Contrary to LSH, model words are extracted from the product description instead of the product title. The data is cleaned as previously delineated, but as we do not utilize a list anymore and compare key-value pairs instead, no brands are added. Finally, when dissimilarities between all candidate-pairs from LSH are calculated and a dissimilarity matrix is obtained, MSM uses an adapted hierarchical single linkage clustering on this matrix. The dissimilarities of two products in the matrix reflect the the distance between clusters, such that clusters with the smallest distance are merged until this distance surpasses a predefined threshold value ϵ. If a cluster contains multiple products, these products are regarded as final duplicates, such that the performance of the MSM can now be evaluated.

3.4 Evaluation Methods

To evaluate the performance of the MSMPb method, I calculate the F_1- and F_1^*-measures: $F_1 = \frac{2*Precision*Recall}{Precision+Recall}$ and $F_1^* = \frac{2*PQ*PC}{PQ+PC}$, where pair quality and pair completeness are calculated as $PQ = \frac{D_f}{N_c}$ and $PC = \frac{D_f}{D_n}$, respectively. Here, D_f represents the amount of duplicates found, D_c the total amount of duplicates and N_c the number of made comparisons. The values for PQ and PC among others, can change for different fractions of comparisons, which is N_c divided by the total number of possible comparisons. Hence, the PQ and PC need to be plotted against the fractions of comparisons to be able to evaluate to which extend these changes take place.

4 Evaluation

The objective of this paper is to find duplicate television products in the aforementioned dataset with a higher accuracy than the method from [12], while maintaining the number of comparisons low. LSH has been implemented to achieve the latter. As data cleaning has been applied to the MSMPb, this also needs to be done for the MSMP - which had no data cleaning - to isolate the effect of the brands on the method's performance.

In order to retrieve robust results, the bootstrapping method is employed. As training data, approximately 63% of the original data is used for each of the 5 utilized bootstraps. Consequently, the remaining data is considered as the test data. The LSH performance is evaluated firstly, after which the MSM performance is analyzed.

4.1 LSH Performance

As the signature matrix consists of n=800 rows, I evaluate the performance of LSH on the F_1^*-measure for b=[20, 25, 40, 50, 80, 100, 125, 200, 400]. To optimize the F_1^*-measure, a trade-off between the PQ and PC needs to be found. Different values of PQ and PC are plotted against the fraction of comparisons in Figures 1 and 2. The fraction of comparisons represents the ratio of candidate duplicate pairs obtained from LSH to the total number of possible comparisons and depends on the value of t. As this fraction increases, PQ decreases, while PC rises. The optimal trade-off with the highest F_1^*-measure, therefore lies somewhere in between. The same holds true for the F_1-measure, when taking precision and recall into account. The best measure is found after calculating the measure for all different bands. A threshold of $t = 0.83$ turns out to perform the best in the training data, corresponding to $b = 40$ and $r = 20$. Hence, these parameters will be used for the evaluation of the test data for the MSMPb.

4.2 MSM Performance

Due to the lack of packages for hierarchical clustering in R, a sub-optimal algorithm is employed. As a result running times would become very high despite

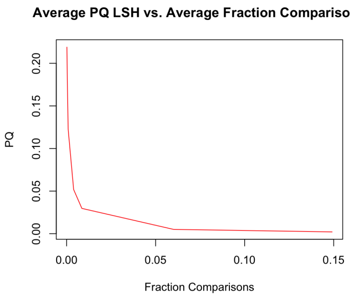

Fig. 1. Pair quality for different fraction of comparisons

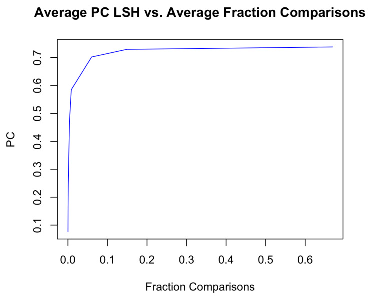

Fig. 2. Pair completeness for different fraction of comparisons

the pre-selection method, which is why I have used the optimal parameters $[\alpha = 0.602, \beta = 0.000, \gamma = 0.756, \mu = 0.650]$ from [11]. For the only remaining parameter, ϵ, I perform a grid search ranging from 0 to 1 with a step-size of 0.1. The mean of the best values of ϵ for the 5 bootstraps on the training data is 0.680. This parameter value will be used to calculate the performance measures of the MSMPb and MSMP on the test data. The average results over the 5 bootstraps on the test data are illustrated in Table 4.2 below. The MSMPb has a higher F_1-value of 0.012 difference and also a higher F_1^*-value of 0.0246 difference compared to the MSMP. Put differently, the F_1 value increases with 31.4% and the F_1^*-value grows with 39.5%. Hence, the addition of brands to the model words set in the pre-selection method leads to a higher precision in duplicate detection, while barely affecting the running time.

Method	F_1-**measure**	**Precision**	**Recall**	F_1^*-**measure**	**PQ**	**PC**
MSMPb	0.0509	0.3296	0.0331	0.0868	0.3296	0.0563
MSMP	0.0382	0.2660	0.0207	0.0622	0.2660	0.0356

Table 1. Average performance of the MSMP-variants

5 Conclusion

In this paper I investigate how the inclusion of brand-names to the set of model words in the pre-selection method of Multi-component Similarity Method contributes to the overall performance of finding product duplicates in an efficient way. The method proposed to study this is the MSMPb, which is an extension of the MSMP from [12]. The pre-selection method applied is Locality-Sensitive Hashing. This is where this paper particularly deviates from that of [12]. A larger set of model words is used in this research, while also performing data cleaning on a data set of television product specifications from 4 different Web shops.

On the one hand, the addition of these brands only enlarges the total set of model words for a small fraction, practically unaffecting the total running time. On the other hand, the performance evaluated by the F_1- and F_1^*-measures do increase with a staggering 31.4% and 39.5%, respectively. Thus, the MSMPb outperforms the standard MSMP on both evaluation measures such that the MSMPb is a worthwhile extension of the MSMP.

This paper can be extended by including a grid search over multiple parameters, which may come at the cost of scalability. Differently, the algorithms applied may be optimized (or run in a different program from R) to reduce the running time. Besides, as this paper outperforms the method of [12], which the method from [7] does as well, the two methods could be combined to construct an even more enhanced method.

References

1. Amazon, Inc. `http://www.amazon.com`
2. Best Buy Co., Inc. `https://www.bestbuy.com`
3. Computer Nerds International, Inc. `https://www.thenerds.net`
4. Newegg, Inc. `https://www.newegg.com` (now defunct)
5. Cohen, E., Datar, M., Fujiwara, S., Gionis, A., Indyk, P., Motwani, R., Ullman, J.D., Yang, C.: Finding interesting associations without support pruning. IEEE Transactions on Knowledge and Data Engineering **13**(1), 64–78 (2001)
6. De Bakker, M., Frasincar, F., Vandic, D.: A hybrid model words-driven approach for web product duplicate detection. In: Advanced Information Systems Engineering: 25th International Conference, CAiSE 2013, Valencia, Spain, June 17-21, 2013. Proceedings 25. pp. 149–161. Springer (2013)
7. Hartveld, A., van Keulen, M., Mathol, D., van Noort, T., Plaatsman, T., Frasincar, F., Schouten, K.: An lsh-based model-words-driven product duplicate detection method. In: International Conference on Advanced Information Systems Engineering. pp. 409–423. Springer (2018)
8. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on Theory of computing. pp. 604–613 (1998)
9. Papadakis, G., Ioannou, E., Palpanas, T., Niederée, C., Nejdl, W.: A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE Transactions on Knowledge and Data Engineering **25**(12), 2665–2682 (2012)
10. Slaney, M., Casey, M.: Locality-sensitive hashing for finding nearest neighbors [lecture notes]. IEEE Signal processing magazine **25**(2), 128–131 (2008)
11. Van Bezu, R., Borst, S., Rijkse, R., Verhagen, J., Vandic, D., Frasincar, F.: Multi-component similarity method for web product duplicate detection. In: Proceedings of the 30th annual ACM symposium on applied computing. pp. 761–768 (2015)
12. Van Dam, I., van Ginkel, G., Kuipers, W., Nijenhuis, N., Vandic, D., Frasincar, F.: Duplicate detection in web shops using lsh to reduce the number of computations. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing. pp. 772–779 (2016)
13. Wikipedia contributors: List of television manufacturers. `https://en.wikipedia.org/wiki/List-of-television-manufacturers`

6 Appendix

List of used television brands:
Acer, Admiral, Aiwa, Akai, Alba, Amstrad, Andrea, Apex , Apple , Arcam, Arise India, AGA , Audiovox, AWA, Baird, BangOlufsen, Beko, BenQ, Binatone, Blaupunkt, BPL, Brionvega, Bush, CGE, Changhong, ChiMei, Compal, Conar, Continental, Cossor, Craig, Curtis, Daewoo, Dell, Delmonico, DuMont, Durabrand, Dynatron, English, EKCO, Electrohome, Element , Emerson, EMI, Farnsworth, Ferguson, Ferranti, Finlux, Fisher, Fujitsu, Funai, Geloso, General, GoldStar, Goodmans , Google, Gradiente, Grundig, Haier, Hallicrafters, Hannspree, Heath, Hinari , HMV, Hisense, Hitachi, Hoffman , Itel, ITT , Jensen, JVC, Kenmore, Kent , Kloss , Kogan, Kolster-Brandes, Konka, Lanix, Le.com, LG , Loewe, Luxor, Magnavox, Marantz, Marconiphone, Matsui, Memorex, Micromax, Metz, Mitsubishi, Mivar, Motorola, Muntz , Murphy , NEC, Nokia, Nordmende, Onida, Orion , Packard, Panasonic , Pensonic, Philco , Philips, Pioneer, Planar , Polaroid, ProLine, ProScan, Pye, Pyle, Quasar, RadioShack, Rauland-Borg, RCA, Realistic, Rediffusion, SABA, Salora, Samsung, Sansui, Sanyo, Schneider , Seiki, Sèleco, Setchell , Sharp, Siemens, Skyworth, Sony, Soyo, Stromberg-Carlson, Supersonic, Sylvania, Tandy, Tatung , TCL , Technics, TECO, Teleavia, Telefunken, Teletronics, Thomson , Thorn , Toshiba, TPV , TP , United , Vestel, Videocon, Videoton, Vizio, Vu , Walton, Westinghouse , White-Westinghouse, Xiaomi, Zanussi, Zenith , Zonda.