Finding Top-k Competitors from Large Unstructured Datasets

RAVULAPALLI SOWJANYA[1], SREEDHAR PULIPATI[2]

[1]PG Scholar, Dept of CSE, QIS College of Engineering & Technology, Ongole, AP, India.

[2]Associate Professor, Dept of CSE, QIS College of Engineering & Technology, Ongole, AP, India

ABSTRACT

Data mining is the prevalent region of the analysis which encourages the business development process, for example, mining client preference, mining web data's to get sentiment about the item or services and mining the competitors of a particular business. In the current business situation, there is a need to examine the focused competitive features and factors of a item that most influence its competitiveness. The assessment of competitiveness dependably utilizes the client sentiments as far as reviews, ratings and abundant source of data's from the web and different sources. In this paper, a formal meaning of the competitive mining is describes with its related works. We introduce proficient techniques for evaluating competitiveness in expansive review datasets and address the common issue of finding the top k competitors of a given item. Finally the paper gives the difficulties and significance in the competitor mining works with ideal improvements.

Keywords: Data mining, Web mining, Information Search and Retrieval, Competitor Mining.

I.INTRODUCTION

The key significance of identifying and observing business competitors is an unavoidable research, which encouraged by a few business challenges. Observing and identifying company's competitors have considered in the current work. Information mining is the ideal method for dealing with such enormous data's for mining competitors. Product reviews form online offer rich data about clients' sentiments and enthusiasm to get a general thought with respect to competitors. In any case, it is for the most part hard to see all surveys in various sites for competitive items and obtain insightful recommendations physically. In the prior works in the literary

works, many creators examined such huge client information brilliantly and proficiently. For example, considerable measures of studies about online reviews were expressed to assemble item opinion examination from online reviews in various levels. Our intensity paradigm depends on the following perception: the competitiveness between two products is based on whether they complete for the consideration and business of the same groups of clients. For example, two restaurants that exist in various nations are clearly not competitive, since there is no overlap between their objective groups. Consider the case appeared in Figure 1.

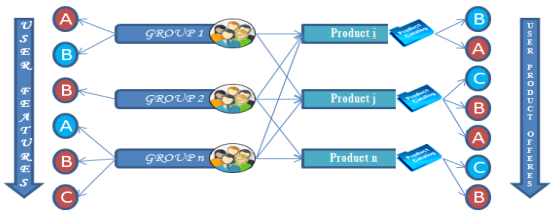

Figure1: An example of our competitiveness paradigm between products

The figure outlines the intensity between three products I, j and k. Every product is mapped to the set of services that it can offer to a client. Three services are considered in this illustration: A, B and C. Despite the fact that this basic illustration thinks about just binary features (i.e. accessible/not accessible), our genuine formalization represents a considerably richer space including binary, categorical and numerical features. The left half of the figure indicates three groups of clients g1, g2, and g3. Each group speaks to an alternate market segment. Clients are grouped in view of their preferences with regard to the features. For instance, the clients in g2 are just inspired by services A and B. We analyze that products I and k are not aggressive, since they just don't request to similar groups of clients. Then again, j completes with both I and k. At last, an interesting perception is that j competes for 4 clients with i and for 9 clients with k. In other words, k is a greater competitor for j, since it guarantees a considerably bigger portion of its market of the overall share than i. This case describes the perfect situation, in which we approach the total set of clients in guaranteed market, and in addition to particular market sections and their necessities. Practically speaking, be that as it may, such data isn't accessible. To beat this, we describe a strategy for processing every one of the sections in a given market in view of mining large review datasets. This technique enables us to operationalize our meaning of competitiveness and address the issue of finding the best k competitors of a product in any given market. As we appear

in our work, this issue presents critical computational difficulties, particularly in the presence of expansive datasets with hundreds or thousands of products, for example, those that are regularly found in standard areas. We address these difficulties by means of a highly scalable system for top-k computation, including a productive evaluation algorithm and appropriate records. Our work makes the accompanying commitments: (a) A formal meaning of the competitiveness between two products, in light of their interest to the different client segments in their market. Our approach overcomes the dependence of past work on scarce comparative proof mined from content. (b) A formal procedure for the recognizable proof of the various kinds of clients in a given market, as well with respect to the estimation of the level of clients that have a place with each type. (c) A highly scalable system for finding the top k competitors of a given product in very high dimensional datasets.

II. RELATED WORK

This section presents the prior work suggested in competitor mining. Authors in developed an automatic system that discovers competing companies from public information sources. In this system data is crawled from text and it uses transformation oriented learning to obtain appropriate data normalization, combines structured and unstructured information sources, uses probabilistic modeling to represent models of linked data, and succeeds in autonomously discovering competitors. Bayesian network for competitor identification technique is used. The authors also introduced the iterative graph reconstruction process for inference in relational data [6], and shown that it leads to improvements in performance. To find the competitors, the authors used machine learning algorithms and probabilistic approaches. They also validate system results and deploy it on the web as a powerful analytic tool for individual and institutional investors. However, the technique has many problems like finding alliances and market demands using the machine learning approach. In the paper [7], authors presented a formal definition of the competitiveness between two items. Authors used many domains and handled many shortcomings of previous works. In this paper, the author considered the position of the items in the multi-dimensional feature space, and the preferences and opinions of the users. However, the technique addressed many problems like finding the top-k competitors of a given item and handling

structured data. Authors in [8] proposed a new online metrics for competitor relationship predicting. This is based on the content, firm links and website log to measure the presence of online isomorphism, here the Competitive isomorphism, which is a phenomenon of competing firms becoming similar as they mimic each other under common market services [9]. Through different analysis they find that predictive models for competitor identification based on online metrics are largely superior to those using offline data. The technique is combined the online and offline metrics to boost the predictive performance. The system also performed the ranking process with the considerations of likelihood. Several works in the same strategy in literature have discussed the need for accurate identification of competitors and provided theoretical frameworks for that. Given the expected isomorphism between competing firms, the process of competitor identification through pair-wise analysis [10] of similarities between focal and target firms is well founded. The unit of analysis is a pair of firms since competitor relationship is seen as a unique interaction between the pair. Authors in [11] have suggested frameworks for manual identification of competitors. The manual nature of these frameworks makes them very costly for competitor identification over a large number of focal and target firms, and over time [12]. In the paper[13], authors attempts to accomplish a novel task of mining competitive information with respect to an entity , the entity such as a company, product or person from the web. The authors proposed an algorithm called “CoMiner”, which first extracts a set of comparative candidates of the input entity and then ranks them according to the comparability, and finally extracts the competitive fields. But the CoMiner [14] specifically developed to support for specific domain. However the effort for the further domains is still challenging. Authors in [15] have proposed ranking methods to give the competitor in a ranked way. They have used data from location based social media. Authors proposed the use of Page-Rank model and it’s variant to obtain the Competitive Rank of firms. However mining competitors from the social media developed many privacy related issues.

III PROBLEM DEFINITION

Many researchers were examining the analyses on product feature extracting information and competitor analysis. The issue of dynamically extracted information records that are identified with the client

given may have two kinds of documents like ordered and unordered structures. Taking care of unstructured dataset in the web repository may dependably make many challenges. This strategy plays out a novel information extraction by means for recognizing the information regions and merging followed by session and request result set reorganization of the records. The extracted information should be changed over into structured one and internal structures are distinguished. Despite the fact that the prior work CMiner++ gives great outcome, despite every product it limits in few cases like area specifications, information handling and dynamic information management issues.

IV.THE CMINER++ ALGORITHM ANALYSIS

We introduce CMiner++, a correct methodology for finding the top k competitors of a given product. Our proposed algorithm makes utilization of the skyline to pyramid all together to reduce the quantity of products that should be considered. Given that we just think about the top k competitors, we can incrementally process the score of every candidate and stop when it is ensured that the top k has emerged. The pseudocode is given in Algorithm 1.The input incorporates the collection of products I, the collection of features F, the product of interest I, the number k of best competitors to recover, the set Q of queries and their probabilities, and the skyline pyramid D_i.

Algorithm 1 CMiner

Input: Set of items $\mathcal{I}$, Item of interest $i \in \mathcal{I}$, feature space $\mathcal{F}$, Collection $\mathcal{Q} \in 2^{\mathcal{F}}$ of queries with non-zero weights, skyline pyramid $\mathcal{D}_{\mathcal{I}}$, int k
Output: Set of top-k competitors for i

```
1: TopK ← masters(i)
2: if ( k ≤ |TopK| ) then
3:     return TopK
4: end if
5: k ← k − |TopK|
6: LB ← −1
7: X ←GETSLAVES(TopK, D_I) ∪ D_I[0]
8: while ( |X| != 0 ) do
9:     X ← UPDATETOPK(k, LB, X)
10:    if ( |X| != 0 ) then
11:        TopK ←MERGE(TopK, X)
12:        if ( |TopK| = k ) then
13:            LB ←WORSTIN(TopK)
14:        end if
15:        X ←GETSLAVES(X, D_I)
16:    end if
17: end while
18: return TopK

19: Routine UPDATETOPK(k, LB, X)
20: localTopK ← ∅
21: low(j) ← 0, ∀j ∈ X.
22: up(j) ← Σ_{q∈Q} p(q) × V^q_{j,j}, ∀j ∈ X.
23: for every q ∈ Q do
24:     maxV ← p(q) × V^q_{i,i}
25:     for every item j ∈ X do
26:         up(j) ← up(j) − maxV + p(q) × V^q_{i,j}
27:         if ( up(j) < LB ) then
28:             X ← X \ {j}
29:         else
30:             low(j) ← low(j) + p(q) × V^q_{j,j}
31:             localTopK.update(j, low(j))
32:             if ( |localTopK| ≥ k ) then
33:                 LB ←WORSTIN(localTopK)
34:             end if
35:         end if
36:     end for
37:     if (|X| ≤ k ) then
38:         break
39:     end if
40: end for
41: for every item j ∈ X do
42:     for every remaining q ∈ Q do
43:         low(j) ← low(j) + p(q) × V^q_{i,j}
44:     end for
45:     localTopK.update(j, low(j))
46: end for
47: return TOPK(localTopK)
```

Figure 2: CMiner++ Pseudocode

The methodology in the first place recovers the products that dominate i, by means of masters (i) (line 1). These products have the most possible competitiveness intensity with i. In the event that at any rate k such products exist, we report those and finish up (lines 2-4). Else, we add them to Top K and decrement our financial plan of k as needs be (line 5). The variable LB keeps up the most reduced lower bound from the current top k set (line 6) and is utilized to prune

competitors. In line 7, we initialize the arrangement of competitors X as the union of products in the start with layer of the pyramid and the set of products dominated by those as of now in the Top K. This is gained by means of calling GETSLAVES (TopK, D_i). In each cycle of lines 8-17, CMiner++ feeds the set of competitors X to the UPDATETOPK () schedule, which prunes products in view of the LB threshold. It at that point refreshes the TopK set by means of the MERGE () work, which distinguishes the products with the most competitiveness from TopK ∪ X. This can be accomplished in linear time, since both X and TopK are arranged. In line 13, the pruning threshold LB is set to the most worst (least) score among the new TopK. At long last, GETSLAVES () is utilized to grow the set of applicants by including products that are overlapped by those in X.

V.ENHANCING THE CMINER++ ALGORITHM

In this area we describe a few enhancements to the CMiner++ two fundamental routines. We implement these changes into an improved algorithm, which we refer to as CMiner++. We incorporate this variant in our experimental evaluation, where we compare its effectiveness and that of CMiner++, and to that of different baselines.

The UPDATETOPK () Technique

Despite the fact that CMiner++ can effectively prune low quality competitors, a major bottleneck inside the UPDATETOPK () procedure is the calculation of the last competitiveness score between every applicant and the product of interest I (lines 41-46). Speeding up this calculation can have a huge affect on the effectiveness of our algorithm. Next, we illustrate this with a case. Assume that products are characterized in a 4-dimensional space with various features f1, f2, f3, f4. Without loss of generality statement, we accept that all features are numeric. We additionally consider 3 queries q1 = (f1, f2, f3), q2 = (f2, f3, f4) what's more, q3 = (f2, f4), with probabilities w (q1), w (q2), and w (q3), separately. With a specific end goal to figure the competitiveness between two products I and j, we have to think about all queries also, as per given equation, figure $V_{i,j}^{q1} = V_{i,j}^{f1} \times V_{i,j}^{f2} \times V_{i,j}^{f3}$, $V_{i,j}^{q2} = V_{i,j}^{f2} \times V_{i,j}^{f3} \times V_{i,j}^{f4}$, and $V_{i,j}^{q3} = V_{i,j}^{f2} \times V_{i,j}^{f4}$. Given that the three products incorporate common sequences of variables, we would like to avoid from repeating their computation, when possible. To begin with, we sort all features as indicated by their frequency in the given collection of queries. In our illustration, the request is: f2, f3, f4,

f1. In a specific order, (f2, f3) turns into a typical prefix for q1 and q2, though f2 is a typical prefix for every one of the 3 queries. We at that point manufacture a prefix-tree to guarantee that the calculation of such regular prefixes is just finished once. For example, the calculation of $V_{i,j}^{f2} \times V_{i,j}^{f3}$ is done just once and utilized for both q1 and q2. The tree is utilized as a part of lines 41-46 of CMiner++ to facilitate the calculation of the competitiveness between the product of interest and the rest of the candidates in X . This change is inspired by Huffman encoding, where by frequent symbols (includes for our situation) are nearer to the root, so they are encoded with less bits. Note that Huffman encoding is ideal if the images free of each other, similar to the case in our own setting.

The GETSLAVES () Technique

It is utilized to expand the set of competitors by including the products that are ruled by those in a set (lines 7 and 15). From this time forward, we refer to this as the dominator set. A naive usage would incorporate all products that are commanded by no less than one product in the dominator set. In any case, as expressed in Lemma 1, if a product j is ruled by a product j′, then the intensity of j with any product of interest can't be higher than that of j′. This suggests products that are commanded by the kth best product of the given set will have a competitiveness score lower than the present k-th score and will subsequently not be included into the last outcome. Along these lines, we just need to extend the top k − 1 products and just those that have not been extended as of now during a past iteration. In additionally, the GETSLAVES() strategy can be additionally improved by utilizing the lower bound LB (the score of the k-th best competitor) as takes after: rather than restoring every one of the products that are dominated by those in the dominator set, we just have to think about a dominated product j assuming CF (j, j) > LB. This is because of the way that the competitiveness between I and j is upper-limited by the base scope accomplished by both of the two products (over all queries), i.e., CF (I, j) ≤ min (CF (I, I), CF (j, j)). In this manner, a product with a scope ≤ LB can't replace any of the products in the present TopK.

VI. PERFORMANCE EVALUATION

We describe the experiments that we conducted to evaluate our methodology. All experiments were completed on a desktop with a Quad-Core 3.5GHz Processor and 2GB RAM.

Datasets Collection

Our examinations incorporate four datasets, which were collected for the reasons of this product. The datasets were purposefully selected from various domains to depict the cross domain relevance of our approach. In additionally, the full data on every product in our datasets, we also collected the full collection of item reviews that were accessible on the source site. These reviews were utilized to (1) assess queries probabilities and (2) extract the sentiments of analysts on particular features. The highly cited strategy by Ding et al. is utilized to change over each review to a vector of sentiments, where every sentiment is characterized as a feature polarity combination (e.g. service+, food). The level of reviews on a product that express a positive opinion on a particular element is utilized as the feature's numeric value for that product. We describes to these as sentiment features. Table 4 incorporates clear measurements for each dataset, while a cleared by point description is given bellow.

Convergence of Query Probabilities

We describe the way toward estimating the probability of each query by mining substantial datasets of client opinions. The legitimacy of this approach depends on the supposition that the quantity of available reviews is adequate to consider confident estimates. Next, we collide these reviews as takes after. To begin with, we combine every one of the reviews in each dataset into a one set, sort them by their accommodation date, and split the sorted sequence into fixed sixe segments. We at that point iteratively add segments to the review corpus R considered by Eq. 6 and re-process the likelihood of each query in the extended corpus. The vector of probabilities from the ith cycle is then contrasted and that from the (i−1) Th cycle through the L1distance: the sum of the total differences of relating entries (i.e. the two estimates for a similar query in the two vectors). We apply the procedure for fragments of 25 reviews. The outcomes are appeared in Figure 3.

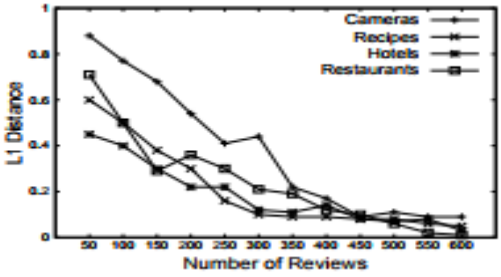

Figure3: Convergence of query probabilities

The x-axis of each plot incorporates the quantity of reviews, while the y-axis is the individual L1 distance. Based on our outcomes, we see that all the datasets showed near indistinguishable patterns. This is an empowering finding with helpful implications, as it educates us that any

conclusions we draw about the joining of the computed probabilities will be applicable crosswise over domains. Second, the figures clearly exhibit the union of the processed probabilities, with the reported L1 distance dropping quickly to inconsequential levels beneath 0.2, after the thought of under 500 reviews. The union of the probabilities is a particularly encouraging result that (I) uncovers a stable absolute distribution for the preferences of the clients over the different queries, and (ii) exhibits that exclusive a little seed of reviews, that is requests of extent smaller than the a great many reviews accessible in each dataset, is adequate to accomplish an exact estimation of the probabilities.

CMiner++ Pruning efficiency

A bit of CMiner++ proficiency originates from its capacity to dispose of or on the other hand specifically evaluate competitors. We show this in Figure 4. The figure incorporates one group of bars for each dataset, with each bar speaking to a various value of k (k ∈ {3, 10, 50, 150, 300}, in the order appeared).

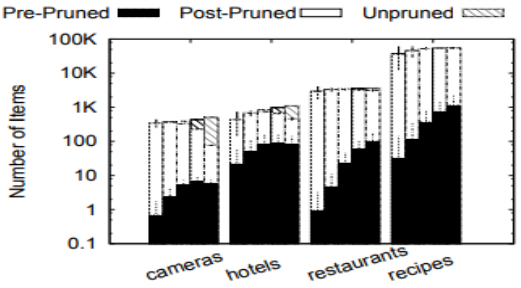

Figure 4: Pruning Effectiveness

The white segment of each bar (post-pruned) speaks to the normal number of products pruned inside UPDATETOPK (). There, a product is pruned if, as we go over the collection of queries Q, its upper bound achieves a value that is lower than LB (the most minimal rival in the present top K). The black segment of each bar (pre-pruned) speaks to the normal number of products that were never added to the candidate set X on the grounds that their most ideal situation (self scope) was apriority more regrettable than LB. Along these lines, they can be disposed of also, we don't need to consider their competitiveness in the setting of the queries. At last, the pattern filled segmentation (unpruned) at the highest point of each bar represents to the normal number of products that were completely assessed in their entirety (i.e. for all queries). We watch that the tremendous larger part of applicants is disposed of by one of the two sorts of pruning that we consider here. The high number of preprinted queries is especially

encouraging, as it suggests the most elevated computational savings. At long last, it is critical to take note of that these discoveries are predictable crosswise over datasets.

VII. CONCLUSION

Data mining has significance with respect to finding the patterns, forecasting, and identification of knowledge and so on. in various business domains. Machine learning methodologies are broadly utilized as a part of different applications. Each business related application employments data mining strategies. To enhance such business or providing suitable competitors for the business to the client require the support of web mining methods. The competitor mining is one such an approach to break down competitors for the selected products. In this paper, we gave a thorough examination of the competitor mining methodologies with its favorable circumstances and disadvantages. At last, the CMiner++ yielded slightest computation time when comparing others. The most important features and process are not considered in the all standard calculations. This can be enhanced in the further researches.

VIII. REFERENCES

[1] Ding, X., Liu, B., Yu, P.S., 2008. A holistic lexicon-based approach to opinion mining. In: Proceedings of the WSDM'08.

[2] Abbasi, A., Chen, H., Salem, A., 2008. Sentiment analysis in multiple languages: feature selection for opinion classification in web forums. ACM Trans. Inf. Syst. 26 (3), 12:1–12:34

[3] Chen, L., Qi, L., Wang, F., 2012. Comparison of feature-level learning methods for mining online consumer reviews. Expert Syst. Appl. 39 (10), 9588–9601.

[4] Zhan, J., Loh, H.T., Liu, Y., 2009. Gather customer concerns from online product reviews – a text summarization approach. Expert Syst. Appl. 36 (2 Part 1), 2107–2115

[5] Jin, Jian, Ping Ji, and Rui Gu. "Identifying comparative customer requirements from product online reviews for competitor analysis." Engineering Applications of Artificial Intelligence 49 (2016): 61-73.

[6] Saxena, Prateek, David Molnar, and Benjamin Livshits. "SCRIPTGARD: automatic context-sensitive sanitization for largescale legacy web applications." Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011.

[7] Ghamisi, Pedram, Jon Atli Benediktsson, and Johannes R. Sveinsson. "Automatic spectral–spatial classification framework based on attribute profiles and supervised feature extraction." IEEE Transactions on Geoscience and Remote Sensing 52.9 (2014): 5771-5782.

[8] Petrucci, Giulio. "Information extraction for learning expressive ontologies." In European Semantic Web Conference, pp. 740-750. Springer, Cham, 2015.

[9] Gentile, Anna Lisa, Ziqi Zhang, Isabelle Augenstein, and Fabio Ciravegna. "Unsupervised wrapper induction using linked data." In Proceedings of the seventh international conference on Knowledge capture, pp. 41-48. ACM, 2013.

[10] K. Simon and G. Lausen, "ViPER: Augmenting Automatic Information Extraction with Visual Perceptions," Proc. 14th ACM Int'l Conf. Information and Knowledge Management, pp. 381-388, 2005.

[11] Zelenko, Dmitry, and Oleg Semin. "Automatic competitor identification from public information sources." International Journal of Computational Intelligence and Applications 2.03 (2002): 287-294.[12] Lappas, Theodoros, George Valkanas, and Dimitrios Gunopulos. "Efficient and domain-invariantcompetitor mining." Proceedings of the 18th ACM SI*GKDD international conference on Knowledge discovery and data mining. ACM, 2012.

[12] Bergen, Mark, and Margaret A. Peteraf. "Competitor identification and competitor analysis: a broad-based managerial approach." Managerial and decision economics 23.4-5 (2002): 157-169.

Auther's Profile:

Ms.RAVULAPALLI SOWJANYA M.Tech Scholar in Computer Science and Engineering , at QIS College Of Engineering and Technology (QISCET),Ongole, India. She has done B.Tech in Computer Scince and Engineering from RISE Gandhi Group Of Institutions, Ongole, India. Her Area Of Research is in Data Mining.

P.SREEDHAR has received his B.Tech in Computer Science and Engineering and M.Tech degree in Computer science and Engineering from JNTU, Hyderabad in 2005 and JNTU, Kakinada in 2010 respectively. He is Persuring his Ph.D. from SSSUTMS, Bhopal. He is dedicated to teaching field from the last 13 years. He has guided 8 P.G and 38 U.G students. His research areas included Data Mining. At present he is working as Associate Professor in QIS College of Engineering &Technology (AUTONOMOUS), Ongole, AP, India.