Interactive Association Map Creation from Documents using Association Rule Mining

Efthimios Mitkousis[1] and Yannis Tzitzikas[1,2]

[1] Computer Science Department, University of Crete, Heraklion, Greece
[2] Information Systems Laboratory, FORTH-ICS, Heraklion, Greece
efthimismitk16@gmail.com, tzitzik@ics.forth.gr

Abstract. One method to aid the understanding of a document corpus, is to try to construct *automatically* a word/term/knowledge map for that corpus by analyzing the contents of the documents. Several methods have been proposed in the literature for this task. In this paper we investigate a novel method that is based on *Association Rule Mining (ARM)*. ARM was proposed for databases, for structured data in general, as a method for data mining, e.g. for market basket analysis. In this paper, we investigate its application over documents. In particular, we leverage association rule mining, through the *Apriori* algorithm, to find pairs of terms that co-occur in documents and their association. Each rule is characterized by its *confidence* and *support*. Then we map these rules to graphical elements. A key merit of the approach is that the user can interactively change the *confidence* and *support* threshold and obtain a different visualization. The evaluation (over small datasets up to datasets with 125.654 distinct words) showed that this approach is feasible and can produce maps that show the dominating words and connections.
Source code: https://github.com/EfthimisM/AssociationMaps
Video: https://youtu.be/eN9VrmmS6Ls

Keywords: Taxonomy Creation · Topic Extraction · Association Rule Mining

1 Introduction

MOTIVATION. In many cases we have to construct a taxonomy/map of words/phrases/entities from a particular corpus of documents, as a means to facilitate the *understanding* of the domain, the understanding of the contents of the corpus, as well as for other tasks including *key topics detection*, production of *summaries and overviews*, detection of *unexpected associations*, etc.

APPROACH. We reduce the problem of map construction to *association rule mining*, and investigate and experimentally analyze various options as regards the words to select, as well as the effect of *confidence* and *support*. The process is illustrated in Figure 1.

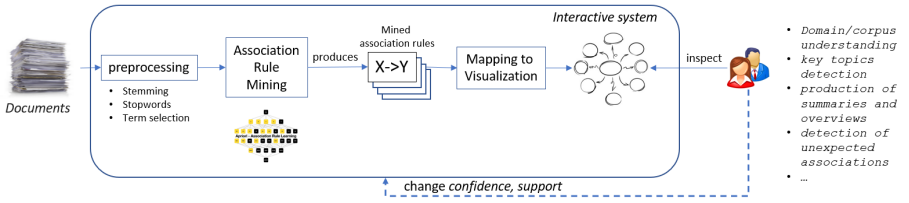

Fig. 1. Overview of Association Rule mining over documents

RELATED WORK and NOVELTY. The problem of automatic construction of lexicons, taxonomies, ontologies, and knowledge graphs is subject of research for many years. Related work include clustering [20, 9], taxonomy creation [16, 13, 12], automated creation of faceted taxonomies [5] methods for automatic *ontology* construction from text (e.g. see the review [3] that captures shallow and deep learning methods), as well as recent methods that use LLMs (Large Language Models) for automatically producing ontologies, e.g. [4]. To the best of our knowledge this is the *first work* that reduces the problem of topic/taxonomy/map creation to association rule mining. We call the produced maps ***Association-Maps***. If applied without any preprocessing or restriction we get an *association word map*. If applied after entity mining, we get an *association knowledge graph*. If applied over a predefined terminology we get an *association taxonomy*. Key emphasis is given on *transparency*, i.e. on enabling the user to interactively change the values of confidence/support, as a means to aid understanding. Another merit is that all the produced edges have a clear interpretation, and the efficiency of this method in comparison to methods that are based on embeddings.

DEMONSTRATION. We shall demonstrate a system that offers this functionality over various datasets and we shall see how confidence and support affects the results. Finally, the paper reports various measurements and findings by testing this method over various datasets and settings.

2 Background: Association Rule Mining

Association Rule Mining (ARM) is a rule-based machine learning method for discovering interesting relations between variables in large databases. Such processes involve detecting frequent itemsets, which are groups of items that commonly appear together, and then creating rules based on these itemsets. Let U be the set of all *objects* (e.g. all products). A *transaction* (e.g. a market basket) is any subset of U. An *association rule* has the form $A \rightarrow B$ where A and B are two disjoint sets of objects ($A, B \subseteq U$, and $A \cap B = \emptyset$). A rule $A \rightarrow B$ is characterized by its *support* and $confidence$ which are metrics to calculate the importance of the given rule. If T is the set of transactions, $I(S)$ denotes the transactions that contain the set of objects in S ($S \subseteq U$), then the support and confidence of a rule $A \rightarrow B$ is defined as: $support(A \rightarrow B) = \frac{|I(A) \cap I(B)|}{|T|}$, i.e. it is the proportion

of T that contain both A and B, while $confidence(A \rightarrow B) = \frac{|I(A) \cap I(B)|}{|I(A)|}$, i.e. it is the proportion of the transactions in $I(A)$ that also contain B. The end goal is to find all possible rules that have high enough $confidence$ and $support$. There are many algorithms for mining association rules, however the most commonly used, and the one we used in our work, is the *Apriori* Algorithm [1]. It works by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database, and this has various applications in domains such as market basket analysis.

3 Mapping to Association Rule Mining

In our case, the notion of item correspond to word(s), and the notion of transaction corresponds to the notion of document. Therefore through association rule mining we can create rules for words and phrases that commonly occur in the same document. The configuration parameters, *support* and *confidence*, are also useful in our setting: the *support* can be used to filter out rules that occur too few times (note that in documents in most of the cases we have a lot of words that occur too few times), while the *confidence* can be used to infer words with high co-occurence as well as taxonomic relationships.

Let $D = \{d_1, \ldots, d_n\}$ be the collection of documents. For a $d \in D$ we shall use $words(d)$ to denote the set of distinct words that appear in d. Let $Words = \cup\{words(d) \mid d \in D\}$. If S is a set, we shall use $P(S)$ to denote the powerset of S, e.g. if $S = \{a, b\}$, then $P(S) = \{\{a, b\}, \{a\}, \{b\}, \emptyset\}$. Let us include a small example to grasp the idea of ARM. Consider the toy collection of documents shown next:

TID	**Items**
T1	An apple is a fruit
T2	Apples are red
T3	Red is a color
T4	Apples are either green or red
T5	Apples are red fruits

At first we remove the *stopwords*, and we perform word *stemming*, ending up with the itemset $U = \{appl, fruit, red, green, color\}$. Their document frequency and occurrences are:

appl (4): T1, T2, T4, T5
fruit (2): T1, T5
red (4): T2, T3, T4, T5
green (1): T4
color (1): T3

If we set the following thresholds: $support = 2$ (as an absolute value, not as a percentage), and $confidence = 0.5$, then we get the following four rules:

• $\{appl\} \rightarrow \{fruit\}$ with confidence = 0.5. This holds because the number of

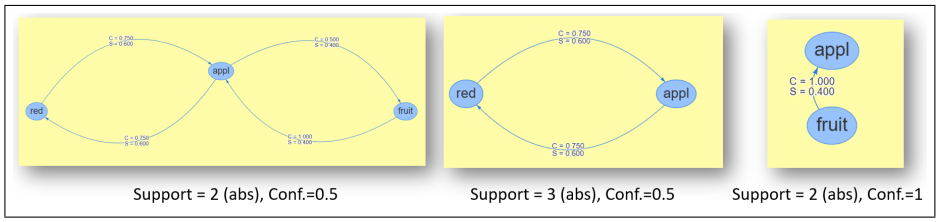

Fig. 2. Mined rules for various support and confidence values.

documents where apple and fruit appear together is 2 (T1, T5) and the number of documents that we find apple is 4 (T1, T2, T4, T5), therefore $conf = 2/4 = 0.5$.
• $\{fruit\} \rightarrow \{appl\}$ with *confidence* $= 1$. This time, since the number of documents where apple and fruit appear together is 2 (T1, T5) and the number of documents that we find fruit is 2 (T1, T5). By using the confidence formula that we showed in §2, $conf = 2/2 = 1$.
• $\{appl\} \rightarrow \{red\}$ with confidence $= 0.75$.
• $\{red\} \rightarrow \{appl\}$ with confidence $= 0.75$.

We can visualize these rules as a network where each term is represented as node and the rules correspond to edges that connect the terms/nodes, as shown in Figure 2 (left). If we increase the support threshold to 3 and keep the confidence to 0.5 we get different results. The only rules that satisfy these thresholds now are:
• $\{appl\} \rightarrow \{red\}$ with confidence $= 0.75$.
• $\{red\} \rightarrow \{appl\}$ with confidence $= 0.75$.
as shown in Figure 2 (middle). In general, the more we increase the support and the confidence, the less rules are mined. If we set the support back to 2, and increase the confidence threshold to 1, we get only $\{fruit\} \rightarrow \{appl\}$ with confidence $= 1$, and thus the network shown in Figure 2 (right).

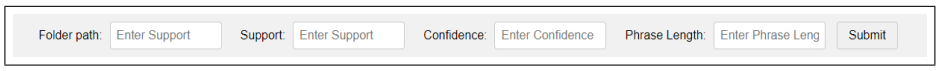

Fig. 3. User input for interactive association maps

4 The Workflow

The process is illustrated in Figure 1. We takes as input a set of documents, and we perform the following steps.
1. We tokenize, we remove stopwords, and apply stemming. Note that on demand one could also perform *term selection*, i.e. define criteria for the words to be considered (either on statistical measures or through a predefined list of keywords of interest).

2. The system takes as input from the user (from a simple GUI shown in Figure 3): (a) a *support* threshold (absolute or percentage), (b) a *confidence* threshold, and (c) the desired *phrase length*. The latter determines what the program considers as terms for indexing and rule extraction. For phrase length of 1, the system treats every individual word as a term. For a phrase length of 2, it considers consecutive pairs of words as terms, and so on.
3. The A-priori algorithm [11] is applied. As mentioned earlier, it is a widely used method for association rule mining. It takes a collection of items and uses specified values for *support* and $confidence$ to identify the largest possible subsets that meet these criteria. However, Apriori is known to be resource-intensive in terms of memory usage. To mitigate potential memory issues, our tool first calculates the number of possible subsets that would need to be stored at each step (powerset). If the number is too large for the available system resources, the process ends and the program returns all the rules that it has extracted up to that point, and the user is prompted to increase the *support* value. This is not problematic for the problem at hand, in the sense that only rules with considerable support value make sense and are useful to mine and visualize.
4. After having identified the subsets of terms that meet our specified parameters, we extract all possible association rules from these subsets and calculate the confidence value for each rule. Any duplicate rules generated during this process are identified and removed to ensure uniqueness.
5. The rules that we found from the extraction now should be stored in a way so every subset found is stored only once with all the associated words and their given $confidence$ value. This step is necessary for the next step so every word has its very own unique node in the graph and duplicates are avoided. So, in order to store the rule $A \rightarrow B$ with $confidence$ x we create an object with a String to hold the value of A, a String to hold the value of B and a float number to contain the value of the confidence.
6. We produce a visualization that takes the form of graph that illustrates the connections between different nodes derived from the previous steps. Each term that has passed all filtering stages and is involved in at least one rule is displayed in the main panel of the screen. This representation provides an intuitive way to explore the associations and hierarchical relationships among the terms.

5 Application Examples

Efficiency. In general, the worst case time complexity depends on the number of distinct words (which is bounded, recall the Heap's Law [7]) and not on the size of the collection. However, the practical runtime is much lower due to the selective generation of subsets of length k and the early termination of the recursive process when subsets fail to meet the *support* threshold.

To check efficiency we performed a number of experiments over a laptop equipped with an AMD Ryzen 7 2700X CPU and 32GB of RAM. However, the program was restricted to using a maximum of 10GB of memory to simulate constrained computational environments.

A few indicative results are shown in Table 1. The first column shows the number of documents, the second the number of unique words, the third the Support, the fourth the Confidence, then the Execution time, the main memory used, MD stands for max depth (it refers to the maximum number of iterations the algorithm needs for finding valid subsets given the support threshold), and the number of extracted rules.

#Docs	Uniq. words	Support	Confidence	Exec. time	Memory	MD	# Rules
40	3.690	0.7	1	0.8"	380MB	7	3
40	3.690	0.7	0.5	0.8"	441MB	7	30
40	3.690	0.55	1	1"	516MB	12	32
40	3.690	0.55	0.5	1"	517MB	12	190
453	125.654	0.93	1	38"	1.2GB	8	1
453	125.654	0.93	0.5	38"	1.2GB	8	112
453	125.654	0.8	1	82"	5.7GB	4	44
453	125.654	0.8	0.5	84"	6GB	4	5400

Table 1. Experimental results about execution time, main memory and extracted rules

The first 4 rows correspond to a *small dataset* that comprises 40 AI-generated descriptions of movies, each having approximately 600 words, in total 3.690 unique words. The last 4 rows, correspond to a *corpus of scientific papers* about *ecosystem restoration* selected by FAO UN, that contains more than 125K unique words (the number of unique words is quite high, the Oxford dictionary has around 500K words, recall Heap's Law). We can see how the confidence/support values determine the number of rules extracted: from 1 to 5400 rules. As regards efficiency, we observe (see the last row) that the maximum time required (without any special optimization or special hardware) is only 84 seconds for mining 5400 rules! (this includes the time for reading the documents).

Examples of maps produced. Two examples of maps created over the small dataset are shown in Figure 4(left). Over the bigger dataset, with the scientific papers, if we use support=0.93 and confidence 1.0, we get only one rule: $\{area\} \rightarrow \{restoration\}$ which is actually the theme of this collection! If we set support=0.65 and confidence: 0.5 we get the map shown in Figure 4(right). which provides more detailed view of the topics of this collection and has an hierarchical structure (the upper level contains general terms like: ecosystem, restoration, effects).

Rule Mining Over Predefined Terms. Over the dataset with the scientific papers, we tested association rule mining restricted on a particular vocabulary. In brief, we used a vocabulary that comprises 272 terms in total: it contains (a) 205 terms that describe *ecosystems*, and (b) 66 terms that describe *restoration types*. As regards (a), this set of terms is constructed from the IUCN global ecosystem typology [8]. It is a hierarchical classification system that in its upper levels, defines ecosystems by their convergent ecological functions and in its lower levels, distinguishes ecosystems with contradicting assemblages of species

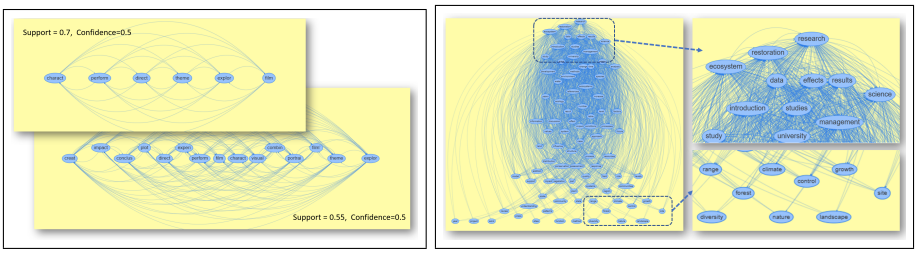

Fig. 4. Left: Examples of the small dataset, Right: Examples over the bigger dataset

engaged in those functions. As regards (b), i.e. *ecosystem restoration types*, this list has been based on a glossary of terms[3], prepared by the Society for Ecological Restoration (SER) [19]. The final version of the vocabulary underwent various edits and refinements from people from FAO. The current version of this vocabulary consists of 67 distinct terms. Figure 5 shows two screenshots of the mined rules. The first enables us to understand which are the main topics, while the second provides a very detailed view of the terms.

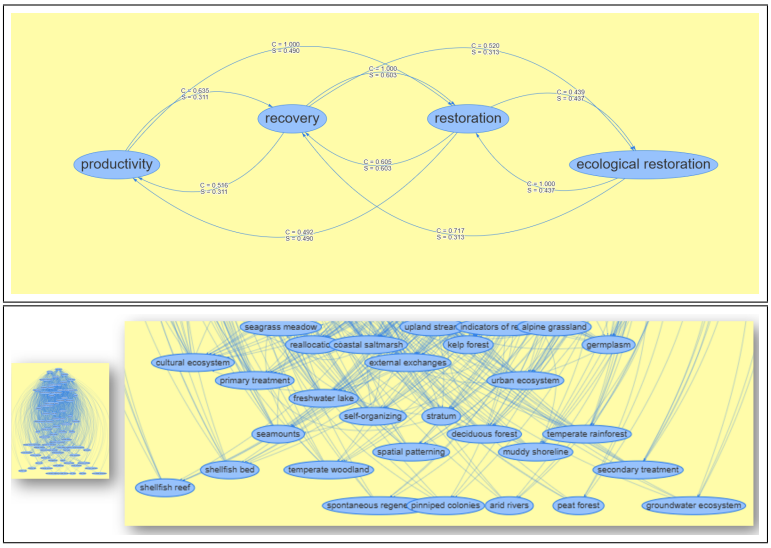

Fig. 5. Examples over the bigger dataset using a controlled vocabulary

Analysis. To understand how confidence and support affect the number of mined rules, Figure 6 shows a plot that shows how many rules (in logscale) are mined for support=0.6, 0.65 and 0.725 and various confidence levels. In these experiments we kept only noun words and we did not apply stemming.

[3] https://www.seraustralasia.com/standards/glossary.html

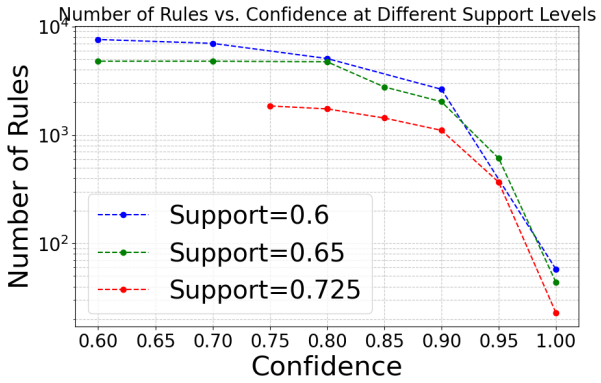

Fig. 6. Number of mined rules for various support and confidence pairs

6 Conclusion and Potential

We have proposed a method for creating what we call *AssociationMaps*, from document collections using association rule mining. A key feature is its interactivity, allowing users to dynamically adjust the *support* and *confidence* thresholds as well as *phrase size*, to tailor the results to their needs. We showcased the feasibility of this approach over various datasets. The *value* of such maps is that they could aid users in understanding a set of documents and getting various overviews of variable granularity. Therefore such maps could be added to any digital library system.

There are several directions that are worth further research. One concerns *visualization and abstraction*, i.e. (i) investigate novel *visualization methods* (none of the current visualization methods [6] seem appropriate for our task, (ii) apply link analysis over the rules (if they are numerous) in order to be able *to reveal the more important* ones (as in [17]),

Another direction is to elaborate on the *granularity of analysis*, i.e. one could consider as transaction not an entire document but smaller units such as sentences or *paragraphs* (like the chunks of RAGs [2]) and then investigate how such a choice affects computational complexity and rule relevance.

The last direction concerns *real-time efficiency*. Note that such maps could enrich *faceted search* interfaces [15, 18], and just like [10] offers geographic maps during the faceted interaction, AssociationMaps could summarize the contents of the current set of documents. during the interaction. In such scenario, we need an effective *caching scheme*: A plain cache could be enough if the corpus is stable, however if the corpus is dynamic (e.g. in the context of interactive search), then a more sophisticated caching scheme is required that can exploit cached *partial results*, analogous to the case of [14]).

References

1. Agrawal, R.: Fast algorithms for mining association rules. VLDB (1994)

2. et al., Y.G.: Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997 **2** (2023)
3. Al-Aswadi, F.N., Chan, H.Y., Gan, K.H.: Automatic ontology construction from text: a review from shallow to deep learning trend. Artificial Intelligence Review **53**(6), 3901–3928 (2020)
4. Babaei Giglou, H., D'Souza, J., Auer, S.: LLMs4OL: Large language models for ontology learning. In: International Semantic Web Conference. pp. 408–427. Springer (2023)
5. Dakka, W., Ipeirotis, P.G., Wood, K.R.: Automatic construction of multifaceted browsing interfaces. In: Proceedings of the 14th ACM international conference on Information and knowledge management. pp. 768–775 (2005)
6. Fister Jr, I., Fister, I., Fister, D., Podgorelec, V., Salcedo-Sanz, S.: A comprehensive review of visualization methods for association rule mining: Taxonomy, challenges, open problems and future ideas. Expert Systems with Applications **233**, 120901 (2023)
7. Heaps, H.S.: Information retrieval: Computational and theoretical aspects. Academic Press, Inc. (1978)
8. Keith, D.A., Ferrer-Paris, J.R., Nicholson, E., Kingsford, R.T.: IUCN global ecosystem typology 2.0. Descriptive profiles for biomes and ecosystem functional groups. IUCN, Gland (2020)
9. Kopidaki, S., Papadakos, P., Tzitzikas, Y.: STC+ and NM-STC: Two novel online results clustering methods for web searching. In: Web Information Systems Engineering-WISE 2009. pp. 523–537. Springer (2009)
10. Lionakis, P., Tzitzikas, Y.: PFSgeo: Preference-enriched faceted search for geographical data. In: On the Move to Meaningful Internet Systems. OTM 2017 Conferences: Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, October 23-27, 2017, Proceedings, Part II. pp. 125–143. Springer (2017)
11. Liu, Y.: Study on application of apriori algorithm in data mining. In: 2010 Second international conference on computer modeling and simulation. vol. 3, pp. 111–114. IEEE (2010)
12. Luu, A.T.: Automatic taxonomy construction from textual documents. Ph.D. thesis (2017)
13. Medelyan, O., Witten, I.H., Divoli, A., Broekstra, J.: Automatic construction of lexicons, taxonomies, ontologies, and other knowledge structures. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **3**(4), 257–279 (2013)
14. Papadakis, M., Tzitzikas, Y.: Answering keyword queries through cached subqueries in best match retrieval models. Journal of Intelligent Information Systems **44**, 67–106 (2015)
15. Sacco, G.M., Tzitzikas, Y., et al.: Dynamic taxonomies and faceted search: theory, practice, and experience, vol. 25. Springer (2009)
16. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. pp. 206–213 (1999)
17. Tzitzikas, Y., Hainaut, J.: How to tame a very large ER diagram (using link analysis and force-directed drawing algorithms). In: International Conference on Conceptual Modeling. pp. 144–159. Springer (2005)
18. Tzitzikas, Y., Manolis, N., Papadakos, P.: Faceted exploration of RDF/S datasets: a survey. Journal of Intelligent Information Systems **48**, 329–364 (2017)
19. Whisenant, S.: The society for ecological restoration. Ecological Restoration **29**(3), 207–208 (2011)

20. Zamir, O., Etzioni, O.: Web document clustering: A feasibility demonstration. In: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 46–54 (1998)