ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 7, No. 5, 178-192 (2022)

www.astesj.com

Special Issue on Innovation in Computing, Engineering Science & Technology

ASTESJ

ISSN: 2415-6698

Association Rules for Knowledge Discovery From E-News Articles: A Review of Apriori and FP-Growth Algorithms

Thilini Lakshika*, Amitha Caldera

Department of Information Systems Engineering, University of Colombo School of Computing, Colombo, 00700, Sri Lanka

ARTICLE INFO

Article history:
Received: 15 July, 2022
Accepted: 12 October, 2022
Online: 31 October, 2022

Keywords:
Association Rule Mining
Apriori algorithm
FP-Growth algorithm
Frequent Pattern Mining
Knowledge Discovery

ABSTRACT

Owing to technological development, the internet has become the world's largest platform where an unaccountable amount of e-news information is freely available to use. Most of the time, e-newspaper readers have to examine the massive collection of e-news articles to locate necessary information relevant to them. Massive semi-structured and unstructured texts usually mislead the readers when they search and understand data for some knowledge. Furthermore, manually reading a collection of e-news articles for some knowledge is tedious and unproductive. The literature related to Knowledge Discovery from text documents has had a substantial improvement in this regard and Association Rule Extraction using text documents, in particular, has become a more frequent and imperative research approach to finding out the most significant information, patterns, and features in the text documents while diminishing the time for reading all the documents. This study provides a comprehensive review of Association Rule extraction using textual data covering the essential topics; Pre-processing, steps in Association Rule Mining, and rule mining algorithms. Out of the various existing association rule mining algorithms, the two most important algorithms, Apriori and FP Growth, are chosen for the experiment using e-news articles. Based on the experimental results, this study discusses the performance, significant bottlenecks, recent breakthroughs of rule mining algorithms, and finally the perspective directions to facilitate future research.

1. Introduction

Due to the rapid growth of web technologies and data repositories, 80% of the world's data is now available in electronic format [1,2]. The internet is the most common platform where an enormous amount of semi-structured and unstructured texts is freely available in reports, news articles, web pages, log files, data transaction files, and other types of electronic documents to satisfy human information desires. The escalating nature of the internet lets many documents first appear online before printing, which is a major reason for the availability of a valuable source of knowledge. Hence, information in the printed medium is slowly losing its gleam. In contrast, the information in the electronic medium continues to grow with the easy availability of quick and obsolete data to the world with only a few snaps.

Due to the overloaded information on news websites, it is tough and tedious for e-news readers to remember all important information to get some knowledge to satisfy their information desires [1–4]. Even searching the internet for exact details is time-consuming. The resultant information may confuse users to deal with it due to both relevant and irrelevant information. Extracting knowledge from e-news articles in semi-structured and unstructured formats becomes more arduous, but it is worthwhile to explore those because of the necessity of obtaining knowledge. But, manual analysis of bulky information available in e-news articles for effective extraction of useful information is not conceivable [3]. Similarly, manual knowledge discovery has a very little guarantee of generating enough details to support decision-making compared to automated knowledge discovery [3]. This issue leads e-news article readers to make decisions based on applications in knowledge discovery.

An automatic text analysis tool based on techniques such as Knowledge Discovery, called Text Mining or Text Data Mining from textual data sources, is the most demanding solution to address this problem [1, 2]. The higher availability of massive amounts of electronic documents has been a great motivation for

*Corresponding Author: Thilini Lakshika, Department of Information Systems Engineering, University of Colombo School of Computing, Colombo, 00700, Sri Lanka, +94713179029, tlv@ucsc.cmb.ac.lk

automatically introducing new knowledge acquisition approaches. Hence, the automatic extraction of knowledge from a large number of e-news articles available on the web using Text Mining (TM) concepts has become an increasingly important research topic [1-3].

Data Mining (DM) is the process of extracting, exploring, and analysing large blocks of information to gather meaningful patterns and trends that can consider knowledge in the data sources. Applying DM techniques to extract interesting and useful knowledge is challenging because of the information overloading, higher duplication rates, and ambiguities of data [5]. DM in the text is known as Text Mining (TM) and is generally used to extract meaningful patterns in a collection of text data using Natural Language Processing (NLP) techniques such as Information Extraction (IE) to transform unstructured data into a structured format [6, 7]. One of the main operations in TM is frequent itemset mining for identifying frequent patterns, hidden patterns, themes, and the context in large datasets that are very easy to understand and interpret by data analysts and normal users [1–5]. Frequent itemset mining plays an essential role in many data mining tasks. Algorithms such as Apriori and FP-Growth are the more frequently used Association Rule Mining (ARM) algorithms for mining sequential and emerging patterns.

Researchers typically use data mining techniques such as classification, clustering, regression analysis, and ARM to discover knowledge from massive data sets by applying novel experimental approaches. TM is a precious and beneficial process for organizations with large textual datasets as it supports expanding the decision-making process of organizations, leading to better business outcomes. However, the richness and vagueness of natural language is the most complex challenge in applying TM techniques [2].

So, the research questions (RQs) that we aim to cover with the study are:

- RQ1: What tasks are currently being solved with association rules using textual data?
- RQ2: What areas of applications currently have been addressed with association rules using textual data?
- RQ3: Which pre-processing steps need to be applied for ARM using textual data?
- RQ4: Which ARM algorithm: Apriori or FP Growth has fast execution in association rule generation using e-news articles?
- RQ5: What are the current trends and future problems to be faced by Apriori and FP Growth algorithms in textual data?

The main contributions of this study are,

- Discuss the diverse areas of ARM along with its applications and motivate research on TM.
- A detailed discussion of the main phases in ARM including the most important text pre-processing steps.
- An experiment conducted using the major rule generation algorithms including Apriori and FP-Growth, their usage, performance, and limitations in TM using a collection of e-news articles.
- Discuss the Apriori and FP-Growth algorithm perspective directions to facilitate future research in TM.

The organization of this paper is as follows. Overview in section 2 explores the fundamental ideas related to ARM. Further elaborations on the role of Association Rules, including Apriori and FP-Growth algorithms in Text applications, are presented in section 3. The significant phases in ARM are discussed in section 4. Our experimental application of Apriori and FP Growth algorithms for association rule generation using e-news articles is described in section 5. Section 6 includes the results and discussion, and the directions for future research are presented in section 7. Finally, section 8 concludes the research work.

2. Overview

2.1. Introduction to Data Mining

The automatic knowledge discovery from text documents using TM concepts has become an emerging research topic [1-3]. Extracting, exploring, and analyzing information to discover meaningful patterns and trends in data is the main focus of DM approaches. Even though the DM supports different techniques such as ARM, frequent itemset mining, and pattern mining, applying those techniques for the extraction of interesting and useful knowledge is challenging because of the information overloading, higher duplication rates, and ambiguities of data [5]. The most common issues in DM are handling many patterns, finding relationships between keywords, and generating knowledge using keyword relationships. Typically, DM techniques discover a rich set of patterns that can consider knowledge in the data sources. However, effective analysis of such discovered patterns for making decisions becomes more complicated due to the meaningless patterns and noises present in the discovered patterns. Even though data mining techniques directly deal with structured data, they may be adapted to mine unstructured text data that presents unique characteristics [2].

2.2. Introduction to Text Mining

Text Mining or Document Mining is an emerging field of research due to the need to retrieve important non-trivial information or knowledge and features by transforming a collection of unstructured text data into a structured format [3, 5, 6]. Data Mining using textual data, also known as Text Mining and is generally used to identify meaningful patterns in a collection of text data using Natural Language Processing (NLP) techniques such as Information Extraction (IE) to transform unstructured data into a structured format [6,7]. TM is generally used to extract semantic logic in the textual data while DM is used to discover novel insights and hidden patterns in text data. Even though many DM tools and applications are designed to work on structured data from databases, many TM tools and applications are designed to work with both unstructured and semi-structured databases such as full-text documents, emails, and HTML files. The lack of explicit structure in the unstructured text data

increases the difficulties of discovering the documents' implicit knowledge. Natural language may express the same concept in multiple ways, making it harder to extract and represent the conceptual concepts in the natural text [1].

Hence, text data mining approaches are still attracting more attention in designing novel TM models to retrieve the users' expected data [6]. Association rule discovery is one of the main techniques in TM, and existing rule discovery approaches are categorized as sequential patterns, maximal patterns, and closed patterns [5]. The various applications such as information extraction, summarization, document classification, and document clustering are the more commonly used applications in the research field of text data analysis.

2.3. Basic Text Pre-processing steps

Generally, e-news articles comprise information in an unstructured format that cannot be straightly used for further processing and cause inaccurate results in text mining [3]. Thus, mining extensive e-news article collections in unstructured text format to extract valuable patterns requires applying essential pre-processing steps and converting the information into a more appropriate data format than keeping the information in a plain text file [3]. The required pre-processing steps for the text data are more important before- applying any DM techniques. Text pre-processing steps typically compromise transformation, tokenization, filtration of keywords, and stemming and indexing of the keywords [2,3].

Transformation: Information on the web presents in different document formats and structures, which requires converting them into Extensible Markup Language (XML) format that is amenable to further processing [1,2].

Tokenization: The unstructured text formats in e-news articles cause inaccurate results in text mining [3] and are required to convert them into structured formats and proceed with only valid tokens by removing inappropriate information such as special characters, parentheses, commas, etc. Tokenization plays a significant role in increasing accuracy in knowledge discovery.

Filtration: Finding out the relationship between all the keywords in the e-news articles is required to generate accurate knowledge from a collection of e-news articles. When the document is not filtered well, filtration becomes complex. The filtration of stop words and suffixes is essential to find the relationship between the keywords. Manual filtration of e-news articles will take much time, and users may miss track of what they are searching for.

Stemming: Stemming reduces the ambiguous association rule generation due to suffixes in the natural text and further reduces the complexity and extra memory consumption while refining the effectiveness and performance of knowledge discovery systems [3]. Many works of literature [3] use a well-known rule-based stemming algorithm; Porter stemmer since it meets both the satisfaction and effectiveness in attaining more relevant information in the text.

Indexing: It is required to index the text to use the unordered words in the text documents [1,2]. Manually or automatically generated indexes using textual data can be considered the foundation for the knowledge discovery process [2]. However, it is not easy to apply manual indexing to large-scale textual data in e-news articles due to its limitations such as heavy time consumption [2]. To address these issues, automated indexing procedures have been examined as a technique that allows association extraction techniques on large-scale textual data [2,3]. Many pieces of research follow the most widely used frequency-based automated indexing weighting scheme which is known as TF-IDF (Term Frequency - Inverse Document Frequency) to calculate and assign higher weights to differentiate terms in a source document [2,3]. Extracting association rules using text documents are based on keyword features in the text, and correlations between those keyword features are based on the weights calculated using the TF-IDF method [1–8].

3. Role of Association Rules in Text Applications

The continued growth in large databases emphasizes the necessity of improving mining performance and precision. Hence, many studies develop novel mining algorithms, theories, and improvements to existing methods. ARM is an exciting DM research domain and raises the interest of many researchers to develop highly efficient algorithms to mine association rules from text documents. ARM concepts span across many domains as per the nature of the problem and demand.

The medical field is a universal domain that requires excessive effort in terms of knowledge management. Most medical applications are daily producing a bulk of medical data in free text formats, which is a very time-consuming task for medical professionals to read and interpret. Hence, applications of TM are widely used in the biomedical field to automatically extract medical findings in these free-text reports based on functional keywords [6]. In [6], the author integrated the Apriori algorithm to find interesting medical data patterns and easily understand the results.

The application of ARM for Knowledge discovery in the newspaper domain is very useful since newspaper readers can easily discover knowledge from a collection of documents without reading all the documents manually [3]. For example, in recent studies [1,2], the authors describe their experimental results of applying Extracting Association Rules from Text (EART) using a set of medical documents related to the epidemic of H5N1 avian influenza virus taken from the MEDLINE bibliographic database. EART uses XML technology with TF-IDF Information Retrieval (IR) scheme for feature selection and the DM techniques for association rules extraction using web documents. As a solution to the time-consuming manual itemsets generation, EART identifies the most essential word features such as the name of the disease, reported outbreak location, current status, and type of the victim, etc. for use in association rules extraction. EART consists of four phases; (1) structural phase (2) indexing phase (3) TM phase and (4) visualization phase.

The main focus of the EART system [1,2] is on the keywords that appear in the extracted association rules and their statistical

distributions without considering the order in which the words occur in a sentence. The ERAT system uses extracted association rules to identify the relations between features in the text document collection. The EART system analyses the keywords in the extracted association rules from two perspectives. (1) The co-occurrence of a keyword in one sentence in the original text document. (2) The existence of a keyword in one sentence without co-occurrence. The EART system outperforms other systems that extract and evaluate the association rules using the Apriori algorithm [1,2]. Experimentation results in [1,2] accepting the challenge of extracting association rules from multidimensional information presented in web news documents.

Another study in [3] discovers association rules and thereby enables the user to generate knowledge from a collection of web news articles related to diseases such as Cholera and Dengue. However, the process of association rule extraction has an expanding growth and a massive number of rules can be extracted from the database [2]. To enhance the efficiency in discovering knowledge from the large number of association rules based on user queries, this study includes an association rule training system that contains pre-defined details about each disease such as impacts of those diseases, reasons for spreading diseases, locations, and victims. This system supports reliable decisions makings due to the knowledge discovered by identifying significant association rules depending on the keywords in the query supplied.

Due to the comprehensive, up-to-date, and valuable information, which contains many broad and collective types of concepts or features, many pieces of literature [1–3] use web news documents as their experimental datasets. Furthermore, web news can more easily be adapted to TM as it does not require the involvement of a subject matter expert to understand and interpret the text features and concepts present in news documents.

In [7], the author deviated from traditional ARM techniques and presented a fuzzy framework for identifying association rules in a text document. This fuzzy framework consists of a fuzzy extended Boolean model, and generated fuzzy association rules are applied to query refinement in Information Retrieval. The generated fuzzy association rules let users query the text document and refine it by showing them a list of candidate terms. Furthermore, this study presented different procedures to apply fuzzy association rules automatically and semi-automatically.

A recent study in [5] accepts the challenge of handling duplications and ambiguities in text data by explaining the relationship between rough set-based decision rules and association rules. The rough association rule-based approach in their study improves ARM effectiveness. The specific information included in rough association rules distinguishes them from standard association rules. The rough association rule comprises a set of terms and their frequency distributions. It is also feasible to dynamically update these rough association rules to increase the effectiveness of the results [5].

The demand for text classification is increasing with the increasing number of online texts. In [9], the author presents a novel fast algorithm for classifying textual data using the concept of ARM. This algorithm is capable of deriving feature sets from pre-classified text documents. Later, the Naïve Bayes classifier applies to the derived features to get the final output of the classification. The algorithm's accuracy in classifying a new document directly depends on the associated word sets generated from pre-classified text documents and performs better with the nonoverlapping text categories.

Detecting and displaying topics from a group of documents is another ARM application. For example, in [4], the author presented a novel approach that uses the visualization of generated association rules to identify the topics from a collection of documents. This approach [4] extends the Apriori algorithm-based approach described in [10]. The experimental results in [4] have shown a reliable match between extracted topics to those present in the data set.

3.1. Role of Apriori Algorithm in Text Applications

The preliminary concept of the Apriori algorithm was the supermarket shopping cart transactions with a set of frequently purchased items [11]. Apriori is one of the well-recognized ARM algorithms that find the frequent items in transactional data sources [1–12]. Association rule mining or discovery using the traditional Apriori algorithm is the most common approach in retrieving hidden rules in the data sources [1–12] but has to pay with immense resources and time.

In [10], the author implements two extending algorithms to discover association rules that are primarily different from the traditional Apriori algorithm. The evaluation results in [10] show that these algorithms perform better for problems with small itemsets than large ones compared to the Apriori algorithm. Furthermore, their study elaborates on another algorithm; Apriori-hybrid [10] by combining the best features in the initially proposed two algorithms to increase the performance in ARM-based applications. Experiments on the Apriori-hybrid algorithm show that it has a linear relationship with the number of items in an association rule and the number of transactions in the database.

Another study in [11] implements Apriori MSG-P, a modified association rule mining technique to demonstrate rare association rules in an operational databank taken from a hospital. During the evaluation process, they examine different characteristics in support values by applying multiple minimum support value approaches for discovering rare itemsets. Experimental results prove that these rare itemsets are more effective than the traditional Apriori algorithms in identifying widespread knowledge and patient behaviors in their databank.

In [2,3], the authors used GARW (Generating Association Rule based on Weighting Scheme) algorithm to overcome the multiple passes over the chosen data, which is another major problem in the Apriori algorithm. The GARW algorithm scans only an XML file containing all the keywords above the given threshold values and their frequencies in each document to generate large frequent keywords. The execution time of the GARW algorithm is less than the Apriori algorithm [1,2].

In recent research, the algorithms for ARM implement numerous optimization techniques to minimize space utilization

and save time in computing frequent itemsets and their support values. The novel scalable algorithm presented in [8] proposes a solution for the existing problems in frequent itemset mining by partitioning the search space. This [8] algorithm discovers closed frequent itemsets which are the lossless and concise presentation of all the frequent itemsets that can be mined from a transactional database. The adaptation of the bitwise vertical representation of the database and the divide-and-conquer approach makes this a fast and memory-efficient scalable algorithm. Moreover, this algorithm addresses the main problems in generating the same closed itemset multiple times using an effective and memory-efficient pruning technique. Mainly, this algorithm does not require keeping the extracted closed patterns in the main memory and enables each visited partition of the search space to be mined independently in any sequence or parallelly.

3.2. Role of FP-Growth Algorithm in Text Applications

Frequent Pattern Growth (FP-Growth) is a frequently used alternative algorithm for generating Frequent Item Sets in a data set. This algorithm enhances the Apriori algorithm that overcomes many of the major problems associated with the Apriori algorithm [13]. Theoretical research in [14] has proven that the FP-growth algorithm resolves two significant complications of the traditional Apriori algorithm. The experimental results in [14] demonstrate that the FP-growth algorithm has higher frequent pattern mining efficiency, less memory usage, and less CPU utilization than the Apriori algorithm. The FP-Growth algorithm accompanies the divide-and-conquer strategy and does not generate candidates because it uses FP-Tree. This dense tree data structure can compress the original transaction database to extract Frequent itemsets [14]. Frequent pattern mining using the FP-Growth algorithm works very well in many broad applications [13-29], including software bug detection, clustering, classification, and recommender systems, and supports selecting the best patterns while reducing time and cost.

The FP-Growth algorithm is mainly used for mining complicated patterns from graph databases [15]. This algorithm is designed for itemset mining, not for graph mining and it does not mine frequent subgraphs well [15]. Hence, finding the frequent subgraphs that support is greater than the given minimum support using a collection of graphs is one of the main problems of frequent subgraph mining [15]. Thus, it is required to make necessary variations in the algorithm so it can be efficiently used for graph mining.

Many of the existing document mining methods for document clustering are based on the frequency of keywords presented in the document [15]. Such methods identify a document as a vector and keywords with their frequency as its elements. However, a document has no way to possess relationships among the keywords. Hence, these methods are not adequate in representing the document's ideas. Clustering documents using the FP-Growth algorithm for association rule mining is a novel approach that overcomes the issues in existing approaches [15]. In [15], the author used the FP-Growth algorithm to mine text documents, find association rules, check the similarity between generated association rules using clustering, and use those results to determine frequent subgraphs. This study modified the FP-Growth algorithm to find the frequent subgraph with clustering affinity propagation in a graph. The ability to evaluate the large-scale graph paths improves the performance of this modified algorithm.

The performance of frequent pattern mining algorithms is affected by many factors, including the databases' characteristics. The FEM (FP-growth & Eclat Mining) algorithm proposed in [26] takes advantage of both FP-tree and TID-list (transaction ID list) data structures to efficiently mine short and long frequent patterns from datasets. The experimental results of the FEM algorithm show a significant improvement in the performance of mining frequent patterns compared to the FP-Growth algorithm. Another experiment presented in [25] concluded that the Dynamic FP-tree construction/reordering algorithm (DynFP-growth algorithm) presented by them behaves better than the FP-growth algorithm.

This algorithm modified the original structure of the FP-Growth algorithm by replacing the single linked list which was used to link the tree nodes to the header with a doubly-linked list and further adding a master table to the same header. Even though the existing database is getting updated, the generated FP-tree in the DynFP-growth algorithm doesn't rebuild the tree. Instead, the algorithm has to be performed considering both new transactions in the database and the initially generated FP tree. Hence, this approach can provide speedy responses to any queries even on the databases that are being repeatedly updated.

The N Painting-Growth and Painting-Growth algorithms developed in [28] are two different improved versions of the FP-Growth algorithm which use two-item permutation sets to extract association rules. The N Painting-Growth algorithm builds two-item permutation sets while the Painting-Growth algorithm builds an association picture based on the two-item permutation sets to find out association sets of all frequent items and then extract all the frequent itemsets according to the association sets. These algorithms reduce the overhead of scanning the database twice in traditional FP-Growth algorithms up to once while increasing the time efficiency. Furthermore, using two-item permutation sets gives several advantages such as lesser memory consumption, running faster, low complexity, and being easy to maintain.

4. Association Rule Mining (ARM)

Association Rule Mining (ARM) is an imperative research area to discover frequent patterns by highlighting correlations between word features in the texts [13–18]. In general, ARM is a process with three major phases; (1) the Text Pre-processing phase, (2) the Association Rule Mining phase, and finally (3) the Visualization phase.

4.1. Text Pre-processing phase

The massive amount of information available in unstructured text format cannot be directly used for further processing. Mining

large document collections to extract useful patterns requires applying essential pre-processing steps to the source documents and storing the information in source documents in a more appropriate data structure for further processing than a plain text file [3]. Text pre-processing steps typically compromise tokenization, filtration of keywords, and stemming and indexing of the keywords [3] to improve the performance in the ARM phase.

4.1.1. Text Transformation

Information on the web presents in different document formats and structures which requires converting them into a machine-processable format that can be easily processed by computers [1,2]. The Extensible Markup Language (XML) is a metalanguage that is used to display documents on the internet based on a user-defined markup language that is amenable to further processing. Supporting techniques in XML help automate the indexing of documents and thus make machines processable [1]. The ERAT system [2] initially saved the Web news pages as text documents and later transformed them into XML format for further processing.

4.1.2. Tokenization

Generally, unstructured data in web pages cause inaccurate results in TM [3]. Thus, it is required to convert unstructured data formats into structured formats by splitting the text in documents into words or terms and selecting only valid tokens by removing inappropriate information such as parentheses, commas, special characters, etc., using tokenization. Tokenization is crucial in removing irrelevant tokens, as it can reduce unnecessary memory and time consumption in the knowledge discovery process. MySQL data structures [3] and matrixes developed using MATLAB software [6] can be used to store tokenization operation results, making it easier to perform further operations such as filtration and stemming, etc.

4.1.3. Filtration

Finding out the relationship between all the keywords present in the text is required for generating accurate knowledge from the collection of text documents. But finding relationships becomes complex if the document is not filtered well. Due to the morphological richness of language, text documents contain many suffixes and stop words [3] which occur frequently likewise "a ", "the", "so" and so on. The careful disregard of stop words without losing the sentence's meaning is required as they do not add much meaning to a sentence and the suffixes may attach to the same word but with different forms. Without filtering such suffixes and stop words in the input documents, finding relationships in text documents and generating accurate knowledge using keywords may become an inefficient and complex process [3]. Unimportant words such as articles, determiners, pronouns, prepositions, common adverbs, non-informative verbs, and conjunctions get discarded. Only the more significant words from the document content are used for further processing [2,3].

Manual filtration will take much time and may miss the track quickly. EART [2] used a list of stop words that are frequently used in English document scripts. Moreover, the EART [2] system replaces parentheses, commas, and special characters with whitespaces.

4.1.4. Stemming

Stemming is reducing the variant forms of the same word to its word stem or root which is known as a lemma. This process happens by removing a word's prefixes and suffixes [2]. Suffixes are single or sets of letters added at the end of a word to generate different forms of the initial word while prefixes are single or sets of letters added at the beginning of a word to generate different forms of the same word. Applying the process of stemming is vital in applications with Natural Language Understanding (NLU), Natural Language Processing (NLP), and Information Retrieval (IR). Applying stemming operation before retrieving information from the documents is much important as it reduces the size of the dataset and the ambiguous association rules generated due to the presence of suffixes and prefixes, and further reduces the complexity and extra memory consumption while refining the effectiveness and performance of knowledge discovery systems [3].

Several types of stemming algorithms attempt to convert a word into its stem or root form. Still, they differ concerning performance and accuracy. Many works of literature [3] use a well-known rule-based stemming algorithm; Porter stemmer since it meets both the satisfaction and effectiveness in attaining more relevant information in the text. One of the common problems in porter's stemmer is that it does not give precise root words at all times. But still, the Porter algorithm was found to be the best algorithm to perform stemming among diverse stemming algorithms such as Paise and Krovtz [3] that are described in the literature [3]. The studies in [1,2] designed a novel stemming dictionary (lexicon) which is suitable for use in the medical domain.

4.1.5. Indexing

Indexing using a weighting scheme comes after the filtration and stems in the text documents. Manually or automatically generated indexes using textual data can be considered the foundation for the knowledge discovery process [2]. However, it is not easy to apply manual indexing to large-scale textual data due to its limitations such as heavy time consumption [2]. Hence, automated indexing procedures have to be considered in applications that use ARM techniques on a large scale [2,3]. In automated indexing techniques, each text document is represented by a set of keywords known as index terms where each index term is a single word or a phrase whose semantics support representing the central theme in the document [2]. Thus, each index term has varying relevance in representing and expressing the document's

theme in a document collection. A numerical weight is assigned to each index to capture its effectiveness.

Definition and Notations in TF-IDF

Many pieces of research [2,3] on extracting association rules follow the most widely used frequency-based weighting scheme; TF-IDF (Term Frequency - Inverse Document Frequency) to identify and filter the terms assigning higher weights to differentiate terms in a source document.

The Term Frequency (TF) is the count that denotes the frequency of a term (keyword) in a collection of documents. TF can be calculated using (1),

$$(tf)_{i,j} = \sum_{i=j=1}^{n} (Nt_i, d_j) \quad (1)$$

Where Nt_i, d_j represents the number of times the term t_j occurs in a document d_j.

Inverse Document Frequency (IDF) is the count that denotes the frequency of source documents that contains the term (keyword) at least once. IDF can be calculated using (2),

$$(idf)_{i,j} = \sum_{i=j=1}^{n} log_2 \left(\frac{|D|}{Nt_j}\right) \quad (2)$$

Where Ntj denotes the document frequency(df) of the term t_j which is the number of documents included in the document collection D in which t_j occurs at least once. The total number of documents in the document collection is denoted using |D|.

TF-IDF value for each term (keyword) can be calculated using (3),

$$w(i,j) = tfidf(d_i, t_j) = Nt_i, d_j * \sum_{i=j=1}^{n} log_2 \left(\frac{|D|}{Nt_j}\right) \quad (3)$$

$$w(i,j) = tfidf(d_i, t_j) = (tf)_{i,j} * log_2 \left(\frac{|D|}{df_j}\right) \quad (4)$$

The simplified version of (3) is in (4) and it calculates the w(i,j) only if $Nt_i, d_j \geq 1$. When the words that do not appear in D, the $Nt_i, d_j = 0$ and thereby w(i,j) = 0.

The value of the Document Frequency (df) scales the d logarithmically. Hence, the formula $log_2 \left(\frac{|D|}{Nt_j}\right)$ gives the full weight for *tfidf* if a term occurs in only one document and if a term occurs in all the documents, then the value of *tfidf* would get zero.

The two important aspects reveal from the TF-IDF weighting scheme are,

1. By considering the *tf* values, the more often occurring terms in a document can be identified as the representative terms of the content of the document.
2. By considering the *idf* values, a term that occurs in more documents can be identified as less discriminative.

Many applications based on Information Retrieval (IR), Text summarization, and association rule extraction from text documents use TF-IDF to weight terms. In [2], the author applies a statistical relevance-scoring function to collecting web news documents to assign scores for each keyword based on maximal TFIDF. Depending on the user's requirement, the top N terms with the highest TF-IDF values can be occupied as the final list of keywords used in the ARM phase [1–3]. The major advantage of automated indexing based on TF-IDF is that it reduces the indexing cost [2].

4.2. Association Rule Mining (ARM) phase

Association Rule Mining (ARM) technique automatically finds information by extracting association rules which contain relationships such as "one implies the other" or "occur together" associated within a list of keywords in a collection of indexed documents. Following the user's requirement, the high-frequency keywords in the indexed documents are selected to generate association rules.

4.2.1. Definition and Notations in ARM

Given a set of keywords A = {w1, w2,..., wn} and a collection of indexed documents D={d1,d2,..., dm}, where each document d_i is a collection of keywords such that $d_i \subseteq A$. Let W_i be a set of keywords. A document d_i is said to contain W_i if and only if $W_i \subseteq d_i$. The general expression of an association rule is an implication of the form $W_i \Rightarrow W_j$ where W_i (antecedent) and W_j (consequent) are sets of keywords in a collection of indexed documents such that $W_i \subset A$, $W_j \subset A$ and $W_i \cap W_j = \varphi$.

The most significant two measures used in the Association Rule evaluation process are support(s) and confidence(c).

If s% of documents in the collection of indexed documents D contain $W_i \cup W_j$, then the Association rule $W_i \Rightarrow W_j$ is said to have support (W_i, W_j) that can be calculated using (5).

$$Support(W_i W_j) = \frac{Support\ count\ of\ W_i W_j}{Total\ number\ of\ documents\ (D)} \quad (5)$$

If C % of indexed documents in the collection of documents D that contain W_i also contain W_j, then the Association rule $W_i \Rightarrow W_j$ holds in D is said to have confidence (W_i, W_j) that can be calculated using (6).

$$Confidence\ (W_i \backslash W_j) = \frac{Support\ (W_i W_j)}{Support\ (W_i)} \quad (6)$$

The general definition of the above two basic measures depends on how a transaction is defined based on the domain. As an example, a transaction can be the records in the dataset or database, each document in the collection of documents [2].

The two threshold values, minimum support, and minimum confidence are the constraints of support and confidence values that the user establishes. An association rule whose support and confidence values are higher than the user-defined thresholds is said to be a strong association rule.

Association rules extraction is a two-step process [1-7]: First, find frequent itemsets, all the itemsets that have support above

minimum support value. Secondly, use the identified frequent keyword sets in step 1 to generate the association rules by discarding those rules below minimum confidence. The later step in the Association Rule generation process is more straightforward than step one, requiring more time and effort [2].

4.2.2. Apriori Algorithm in ARM

Apriori is one of the well-recognized ARM algorithms which supports finding frequent items in transactional data sources [1,2] using two main phases, where the first phase counts item or keyword occurrences to determine the large 1-itemsets and the second phase (k) is again two-fold. In the first fold, the large itemsets L_{k-1} found in the $(k-1)^{th}$ pass are used to generate the candidate itemsets C_k by the Apriori function. The database is scanned and the support of candidates in C_k is counted in the second fold.

The below steps demonstrate the general workflow in the Apriori algorithm associated with the terms in Table I.

Table 1. Notations Used in The Apriori Algorithm

Term	Description
k-keywordsets	A keyword set with k- keywordsets
L_k	Set of large k- keywordsets that satisfy min_{sup} value (in the k^{th} pass)
C_k	Set of candidate k- keywordsets generated from potentially large k-keywordsets
min_{sup}	User-specified minimum support value
min_{conf}	User-specified minimum confidence value

Step 1: Scan the indexed documents or database and find all their accumulated counts. Select the set of frequent 1-itemsets that satisfy the threshold min_{sup} value which is known as L_k.

Step 2: Uses L_k to find the candidate 2- itemsets (C_{k+1}) set. This is a two-step process where step 1 is to use the L_k found in the k^{th} pass to generate the candidate 2-itemsets C_{k+1}. The second step is to scan the indexed documents or database and calculate each candidate's support in C_{k+1}.

Step 3: Calculate the support count for all the itemsets in C_{k+1} and select the frequent 2-itemsets that satisfy the threshold minsup value known as L_{k+1}.

Step 4: Continues steps 2 and 3 until no new frequent (k+1)-itemsets are found.

Step 5: Find all the association rules that satisfy the threshold $min_{conf.}$ for each frequent itemsets.

Apriori follows an iterative level-wise search approach, where k-itemsets are used to explore (k+1)-itemsets [1,2,9].

As described in Table II, the input to the above algorithm is a transaction database where each entry is a transaction of the form <Transaction_id, Set of Items> which is also known as horizontal data format.

Table 2: Horizontal Data Format

Transaction _id	Set of Items
1	a,b,c,d
2	b,c,d
3	A c
4	c,d

The Apriori algorithm will encounter two problems when the frequencies in a set of items have a great variation.

- If the value of min_{sup} is set too high, then the association rules that involve rare items will not be detected. To find out the association rules that involve both frequent and rare items, the value of min_{sup} has to be set very low. Since the frequent itemsets will be associated with each other in all possible combinations, this may cause the generation of a large number of candidate sets [13,14].
- Apriori requires a repeated scan of the database for calculating the support count for candidate sets [13,14].

The vertical format was applied to solve the above issues arising in the DM field by transforming the horizontal data layout described in Table II into the vertical data layout described in Table III by scanning the data set for mining frequent itemsets [13,29,30]. The vertical data format in Table III shows the 1-itemset and it contains entries like <Item, Set of Transactions containing the item>.

Table 3: Vertical Data Format

Item	Transaction Ids	Frequency
a	1,3	2
b	1,2	2
c	1,2,3,4	4
d	1,2,4	3

Every item in Table II is considered as a candidate 1-itemset in the above transaction. This layout employed an intersection between the Transaction_id lists to calculate the support value of the 'candidate's itemset identified in subsequent steps. After counting their support values, all the candidate itemsets which are lesser than the minimum support are discarded.

The vertical layout is a refinement of the traditional Apriori algorithm because it substantially reduces the number of database scans [13,29,30]. This improvement is because the support count (Frequency) is stored in vertical data format just computing the cardinality of each Transaction Id set. Hence, the mining

efficiency is increased and the processing time is decreased compared to the default horizontal layout.

4.2.3. FP-Growth Algorithm in ARM

The FP-Growth is an efficient algorithm that has been proposed as an alternative to the Apriori algorithm for mining frequent itemsets without generating candidates [19–24]. It stores the database transactions in a tree structure which are denoted as FP-tree [19–24] where every item in a transaction has a linked list going through all other transactions that contain the item. The FP-tree structure uses a vertical and horizontal database layout to set aside the transactional database in the main memory. Every node in the FP-tree has a counter value that maintains the number of transactions shared by the node. All occurrences of an item in the FP-tree are linked with the next occurrence of the respective item in the FP-tree using links. Apart from that, a header table contains each item in the FP- tree and its support value and a link to refer to the first occurrence of the item in the FP tree. All the items in the FP tree are stored in the descending order of the support value. Hence, more frequently occurring items are organized closer to the root node of the FP-tree and such items are more likely to be shared than others.

The FP-Growth algorithm consists of 3 main stages:

- Generation of conditional pattern base
- Generation of conditional FP-Tree
- Searching for the frequent itemsets

Algorithm 1 demonstrates the general workflow in the FP-Growth algorithm that incorporates the above 3 main stages.

Algorithm 1: FP-Growth algorithm

```
Input: FP-Tree
Output: The complete set of frequent patterns
Method: FP-Growth (Tree, null)
Procedure: FP-Growth (Tree, α)
if the Tree consists of a single path P then;
    for each combination (denoted β) of the nodes in the path,
    P do
        generate patterns β ∪α with support = min support of
        nodes in β
else
    for each a_i in the Header of Tree do
        generate patterns β = a_i∪α with support = a_i.support
        construct β's conditional pattern base and FP-Tree,
        Treeβ
    if Treeβ ≠ null then
        CALL FP-growth (Treeβ, β)
end
```

The algorithm first compresses the database into a highly condensed FP-tree structure, avoiding costly database scans [23]. Next, it applies the divide-and-conquer approach to decompose the ARM tasks into smaller ones avoiding candidate generation [23, 24].

The FP-GrowthThe FP-Growth algorithm has overcome one major problem of the Apriori algorithm. It has overcome one major problem the Apriori algorithm does not generate a great number of candidate itemsets [13]. Furthermore, this algorithm transformed the entire text database into an FP-tree that holds all the database's information. Hence it requires only two scans of the database and does not scan it again and again as in Apriori [14–19]. The first scan is to find out the frequent words and their support count and the second scan is to sort out each transaction according to their descending support count. Moreover, searching frequent itemsets in FP-Tree does not use generated candidates as it is done on the Apriori algorithm. Hence, the FP-Growth algorithm becomes faster than the traditional Apriori algorithm [13-29].

4.3. Visualization phase

ARM often results in a massive number of association rules which requires the user to go through all the generated rules to discover the interesting rules. Manually analyzing such a massive number of rules is a time-consuming, tough, and tiring activity [20,21]. These issues can be addressed by interactive visualization of the extracted association rules in a textual or graphical format. Association rule visualization has an extensive history of transforming bulky amounts of data into a better accessible method for analysis. Interactive visualization techniques fall under three important categories named scatter plots, matrix visualization, and graph-based visualization [31]. The main elements used in these visualization categories are shown in Figure 1.

As shown in Figure 1, scatter plots utilize different interest measures such as confidence and support on the x and y-axis, and association rules with similar values for such interest measures are placed adjacent to each other. The graph-based visualization demonstrates how association rules are shared by individual items while matrix visualizations demonstrate association rules that have the same antecedent or consequent in the same column or row. The association rules in Figure 1 are shown in blue color.

The literature introduced different visualization techniques such as scatter plots, double-decker plots, mosaics, and parallel coordinates plots to visualize and analyze association rules [21]. Nevertheless, most of such techniques are still falling short with the presence of a large number of association rules [21,22]. However, the most traditional visualization techniques are not much appropriate for visualizing massive sets of association rules since they generally consist of many unique antecedents and consequents [21].

In [21], the author introduced a novel visualization method, a grouped matrix-based representation, which automatically creates nested groups of association rules by applying the k-means clustering algorithm to handle the massive number of generated rules. The hierarchical structure formed by these nested groups of association rules can be interactively explored down to each rule. This Matrix-based visualization [20,21] technique represents the antecedent and consequent of the association rule on the x and y-axes while any selected interest measure is displayed at the intersection of the x and y-axes of a given rule.

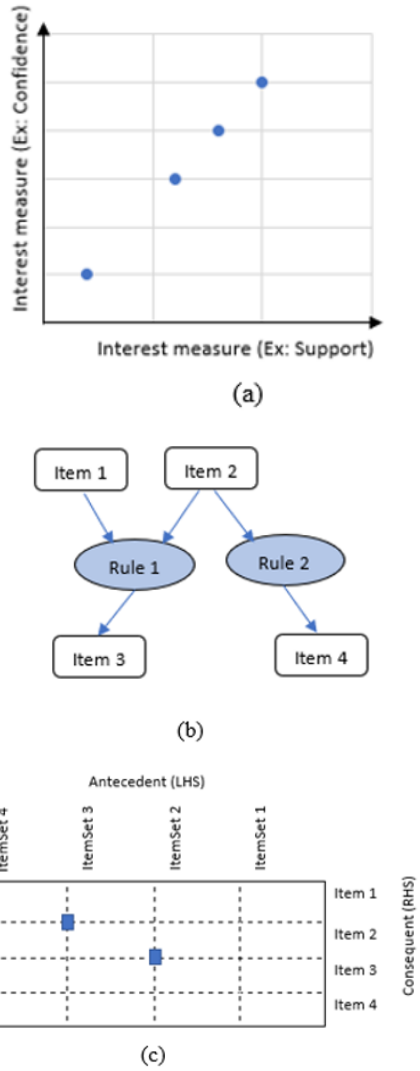

Figure 1. The main elements of association rule visualization using (a) a scatter plot, (b) graph-based visualization, and (c) matrix visualization.

This technique addresses the typical problem that is aroused with huge sets of association rules by grouping antecedents together and letting the end users explore the most interesting groups automatically using different colors and the position of elements in the plot. Grouped matrix-based visualization is easy to understand and unique [21] since most other visualization methods are not able to efficiently deal with huge sets of association rules.

The graph-based techniques [22] visualize the most important extracted association rules using vertices and directed edges where vertices represent each item or itemsets. In contrast, edges represent the relationships in terms of rules. Any selected interest measures are added to the edges of the graph as labels on the edges or as edges with different colors or edges with varying widths of the arrows. These graph-based techniques present a very comprehensible representation of association rules for relatively small sets of association rules selected based on their corresponding confidence score ($min_{conf.}$) value. Graph-based visualization techniques are well suited when the end users are concerned with an aggregated perspective on the most important rules [22]. But this technique offers a clear representation only for relatively small sets of rules, which can be easily chosen based on the selected interest measure.

5. ARM using e-news articles

This section describes the implementation details of the Apriori and FP Growth algorithm for a collection of e-news articles.

5.1. Environment and Dataset

The implementation and testing of Apriori and FP Growth algorithms were done in Python (version 3.7.0) using a workstation with Intel(R) Core (TM) i3-8130U CPU @ 2.20 GHz and 4 GB main memory.

This study evaluated the two algorithms: Apriori and FP Growth using the collection of e-news documents written in the English language released by the Document Understanding Conference (DUC) – 2002 which uses Text REtrieval Conference (TREC) data as the experimental dataset. This dataset compromises 128 topics-related e-news article sets and each article set has between 5 and 15 e-news articles, with an average of 10 articles. The news article must be at least 10 sentences long, but there is no maximum length. The list of topics-related e-news article sets used for the experiment is described in Table IV.

Table 4: The list of topics-related e-news article sets used for the experiment (DUC 2002)

Document set ID	No. of news articles	Total No. of sentences
d113h	5	82
d083a	6	95
d061j	7	124
d063j	8	136
d094c	9	137
d075b	10	183
d070f	11	208
d081a	12	210
d072f	13	229
d069f	14	237

5.2. Methodology

As shown in Figure 2, firstly the basic text pre-processing steps described in section 4.1 including filtration, stemming, and indexing was applied to the collection of e-news articles.

Next, the two algorithms: Apriori and FP Growth were applied to the indexed e-news article collections for generating association rules. We used the TF-IDF numerical statistic to generate index terms from the e-news articles and the count of such generated index terms which are above the threshold value of 0.6 are shown in Table V. All the calculations for identifying index terms were done following (1) to (4) described in the section 4.1.5. Depending on the user's requirement, the top N terms with

the highest TF-IDF values can be occupied as the final list of keywords used in the ARM phase [1–3].

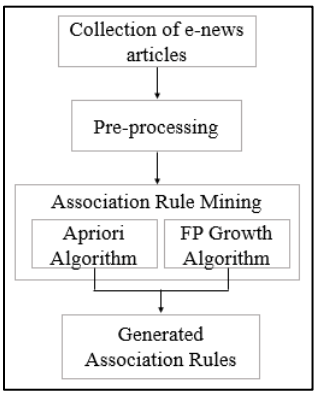

Figure 2. The process of Association Rule Generation

We generated the association rules with different combinations of min_{sup} (values: 60,70,80,90,100) and min_{conf} (values: 80,85,90,95,100) values. All the generated association rules that satisfy each combination of min_{sup} and min_{conf} values were taken for the experiment.

Table 5: The count of index terms (keywords) generated for each topics-related e-news article sets used for the experiment (DUC 2002)

Document set ID	No. Index Terms
d113h	528
d083a	593
d061j	789
d063j	847
d094c	698
d075b	1180
d070f	1271
d081a	1420
d072f	1542
d069f	1782

Table 6: Association Rules generated for the document set ID: d113h using the Apriori algorithm

min_{sup} \ min_{conf}	80	85	90	95	100
60	2909	2221	1912	1459	1127
70	2793	2442	1789	1009	808
80	2543	1992	1405	828	576
90	2046	1551	1079	518	251
100	1506	1119	731	385	73

Our experiment generated the association rules for document sets in Table V using different value combinations of min_{sup} and min_{conf}. Support and Confidence value for each generated association rule were calculated using (5) and (6) in section 4.2.1. All the association rules that satisfy each combination of min_{sup} and min_{conf} values were taken for evaluation.

Table 7: Association Rules generated for the document set ID: d113h using the FP Growth algorithm

min_{sup} \ min_{conf}	80	85	90	95	100
60	2329	2110	1794	1391	978
70	2172	1825	1490	1121	873
80	1856	1513	1201	705	584
90	1621	1297	903	561	240
100	1349	1067	871	378	81

The total number of Association Rules generated using the document set ID: d113h which satisfy the specified min_{sup} and min_{conf} values are listed in Table VI and Table VII.

6. Results & Discussion

The major concern of the experiment described in this study includes a comparison of the execution time of both algorithms for each document set listed in Table V to identify the most effective algorithm for e-news articles.

We monitored the complete execution time taken by each algorithm for generating the set of association rules with different combinations of min_{sup} and min_{conf} values. The execution time taken by each algorithm for generating association rules for the document set ID: d113h with 100% support and 100% confidence is listed in Table VIII.

Table 8: Execution time of Apriori and FP-Growth algorithms for the document set ID: d113h

	Apriori Algorithm		FP Growth Algorithm	
Document set ID	Execution time	No. of rules	Execution time	No. of rules
d113h	40 minutes	73	3 minutes	81
d083a	50 minutes	70	4 minutes	76
d061j	65 minutes	81	5 minutes	80
d063j	70 minutes	85	6 minutes	91
d094c	63 minutes	72	5 minutes	68
d075b	75 minutes	88	6 minutes	76
d070f	87 minutes	89	7 minutes	93
d081a	116 minutes	107	9 minutes	98
d072f	125 minutes	114	11 minutes	91
d069f	146 minutes	123	13 minutes	126

Our implementations of the Apriori and FP Growth algorithms consider each sentence in the e-news article as one database transaction. Hence, each document collection generates a large number of TF-IDF weighted index terms (Table V) or keywords to be considered in association rule generation.

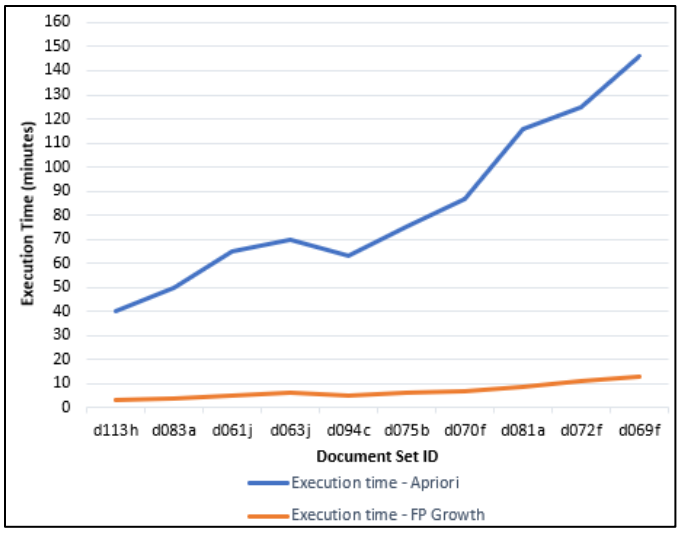

Figure 3: The comparison of algorithm execution time for generating association rules with 100% support and 100% confidence

When the number of generated association rules getting increased, the execution time of both algorithms also getting increased. The details presented in Table VIII and the comparison of execution time in Figure 3 represent that out of these two algorithms, the FP Growth algorithm always performs much better than the Apriori algorithm disregarding the number of generated association rules and several index terms. Mainly this happens due to the major weaknesses in the Apriori algorithm such as several transactional database scans, large memory utilization, and higher execution time.

6.1. The complications in Apriori Algorithm in ARM

Following the experimental results, we evident the below compilations in the Apriori algorithm for association rule generation using textual data.

The Apriori algorithm requires a repeated scan of the database for calculating the support values for each generated candidate set [1-14] which is one of the major problems in the Apriori algorithm. The literature describes the GARW algorithm [1-3] as a solution to this issue since it 'doesn't scan the original documents multiple times as in the Apriori algorithm. Instead, it scans the generated XML file, which includes only the keywords identified during the large frequent keyword sets generation in each document It also includes only the keywords identified during the large frequent keyword sets generation in each document and satisfies the user-defined threshold values. Therefore, the execution time of the GARW algorithm is less than the Apriori algorithm [1,2].

Another major bottleneck in the Apriori algorithm is candidate set generation by handling extremely large candidate sets. Moreover, frequent pattern matching using such massive candidates by searching through each transaction in the database is very expensive and time-consuming. When the frequencies in a set of items have a significant variation and if the value of min_{sup} is set too high, then the association rules that involve infrequent items will not be detected. To identify the association rules that involve both frequent and infrequent items, the value of the min_{sup} has to be set very low. Since each frequent item is getting associated with other frequent items making all possible combinations, this may cause the generation of a massive number of candidate sets [13,14].

As a solution to this issue, the study in [8] presented a fast and memory-efficient scalable algorithm that uses the divide-and-conquer approach. This algorithm uses the bitwise vertical representation of the original database and formalizes the problems that arise in closed itemsets mining by partitioning the search space. This algorithm presents a condensed but lossless representation of all the extracted frequent itemsets in the transactional database and then identifies all the possible closed frequent Itemsets for discovering closed frequent itemsets. Furthermore, this algorithm uses several optimization techniques to reduce space and time utilization in generating itemset closures and their support values. But, one major drawback encountered in this algorithm is the generation of the same closed itemset multiple times [8]. Apart from this algorithm, in [8] the author proposes an effective and efficient pruning technique that does not require storing the set of extracted closed patterns in the main memory. Moreover, this technique allows each visited partition of the search space to be mined parallelly or separately despite the order.

The pruning technique used in the Apriori algorithm does not assure an effective pruning of candidates due to the rapid growth in the frequent itemsets. In this case, the proposed algorithm in [8] ensures that frequent closed itemsets extract all frequent closed itemsets in the transactional database without any duplications while saving memory space and reducing the execution time in itemset closures generation [8].

Association rule generation using the Apriori algorithm considers all the keywords without considering the importance of each keyword. Owing to this, a large collection of association rules will be generated, becoming a tedious and time-consuming task. More importantly, the knowledge discovery using these association rules will not be so worthwhile as it considers all the keywords in the transactional database, including lower-weight keywords [3]. Applying the Apriori algorithm to the vertical data format described in Table III instead of the horizontal data format addresses most of the limitations discussed above [2,3].

6.2. The complications in FP-Growth Algorithm in ARM

Following the experimental results, we evident the below compilations in the FP Growth algorithm for association rule generation using textual data.

The major complication in the FP-Growth algorithm is the cost and time consumption in constructing the FP-Tree. The FP-tree construction in this algorithm has a high requirement for memory space. Hence, the algorithm may not fit in the shared memory when the transactional database gets larger. Apart from that, the observations explained in [25] state that the resulting FP-tree cannot be considered a unique representation of the same logical database that is used in FP-tree generation. But once the construction of the FP-tree is over, it can be easily used to read off the itemsets [19].

The other complication is that the support value can only be calculated after the entire FP-Tree is constructed. Furthermore, when the threshold value of support is getting higher, extensive time is wasted since the pruning can be applied only to one item at once [19,24]. Also, the FP-growth algorithm requires at most two complete database scans [13-29], and scanning the database twice reduces the efficiency of the FP-Growth algorithm.

Even though the Painting-Growth and N Painting-Growth algorithms [28] only scan the database once and are easy to implement, there are still several complications in them. The Painting-Growth algorithm builds the association picture, leading to a large memory overhead. The N Painting-Growth algorithm performs well with macroscale datasets and the implementation of the algorithm is more complex than the Painting-Growth algorithm. Third, these two algorithms repeatedly scan the generated frequent item association sets and extract multi-item frequent sets. This results in reducing the time efficiency of these algorithms [28].

The exponential growth in association rule extraction produces a large number of rules. Association rule extraction approaches are also broadly expanding, and novel algorithms such as Apriori-hybrid are increasing their performance in many real-world applications [6]. Generally, the existing data mining techniques discover innumerable patterns or knowledge from a training data set. Furthermore, noise in the discovered patterns makes the effective knowledge discovery process a big challenge. The concept of closed patterns [8] deals with the noise up to a significant level, but the discovered knowledge still retained many meaningless patterns. Apart from that, the generation and identification of only the interesting patterns based on pre-defined constraints improve the quality of the ARM process [5].

7. Future Directions

Knowledge discovery is an increasing field of research. Still, there is a tremendous need to develop TM approaches using AMR which can guide the user on knowledge discovery in text documents such as web articles identifying important information, and how to generate knowledge. Even though knowledge refinement has been carried out in the literature, the refinements in the ARM are still comparatively less [27].

7.1. Refinements to the Apriori Algorithm

Support and confidence are the two factors considered by the Apriori algorithm to measure the statistical strength of the generated association rules. However, these factors are not useful in deciding the generated rules' convenience or detecting conflicts between the generated rules [27]. Hence, a refinement of the existing factors by introducing a novel factor that considers the convenience of generated rules and identification of conflicts between rules is important for obtaining consistent and interesting patterns.

A few of the major future directions in the Apriori algorithm are,

- Reducing the multiple passes over the chosen data to speed up the algorithm execution.
- Identifying only the most important keywords for generating association rules to reduce generating a large collection of association rules which will not be so meaningful.
- Generating efficient candidate sets by handling exceedingly large candidate sets and matching frequent patterns with a large number of candidates.

Furthermore, the current pruning technique used in the Apriori algorithm does not assure an effective pruning of candidate sets due to the rapid growth in the frequent itemsets.

7.2. Refinements to the FP-Growth Algorithm

The most needed refinement in the FP-Growth algorithm is reducing the cost of development and time consumption in constructing the FP-Tree. Another primary direction in the FP-Growth algorithm is to enhance the algorithm to fit in the shared memory when the transactional database gets larger.

N Painting-Growth and Painting-Growth algorithms developed in [28] provide a direct link for the subsequent ARM research. Enhancing the performance of the Painting-Growth and N Painting-Growth algorithms [28] while increasing the efficiency in infrequent item removal and multi-item frequent sets mining will be some of the leading future works.

8. Conclusion

Effective information extraction using text documents is a massive challenge due to the duplications and ambiguities of data values in the text. The works of literature related to Knowledge Discovery from text documents using Association Rule Extraction have become a more frequent and imperative research approach to finding out the most significant information, patterns, and features in the text documents. This survey provides a comprehensive review of Association Rule extraction using text documents covering a comprehensive review of multiple aspects.

Furthermore, the experimental results discussed in this study answer the question 'Which ARM algorithm: Apriori or FP Growth has fast execution in association rule generation using e-news articles?'. Out of the Apriori and FP Growth algorithms, the FP Growth algorithm always performs much better than the Apriori algorithm disregarding the number of textual contents included in E-news articles. Mainly this happens due to the major weaknesses in the Apriori algorithm such as several transactional database scans, large memory utilization, and higher execution time.

The main outcomes of the study we have conducted are,

- A thorough review of extracting knowledge in text sources using Association Rules.
- A review of the most popular application domains of ARM along with its usage and applications.
- A review of more frequently used association rule extraction algorithms, mainly the Apriori and FP-Growth algorithms in ARM, their usage, recent breakthroughs, advantages, and major bottlenecks in TM.
- Detailed information on the ARM process including a comprehensive review of the Text Pre-processing phase, Association Rule Mining phase, and Visualization phase.
- Detailed information on the perspective directions to facilitate future research.

The major complications in the Apriori algorithm were addressed by introducing a novel compact data structure, FPtree. The FP-growth algorithm that uses the FP tree allows efficient discovery of frequent itemsets and is faster than other ARM algorithms. Even though the value of the support factor influenced the performance of the Apriori algorithm, the performance of the FP-growth algorithm is not influenced by the value of the support factor. Thus, the algorithms with candidate set generation behave well with small databases with the size of a maximum number of transactions of around 50,000 and a support factor of at least 30% [25].

Later, FP-tree-based pattern recognition algorithms such as FEM [26], and DynFP-growth algorithm [25] were developed based on the completeness and compactness of the FP-growth structure. DynFP-growth [25] and FP-growth algorithms that don't require candidate generation perform much better. The experimental results of the FEM [26] and FP-growth [25] algorithms have shown a significant improvement in the performance of mining frequent patterns compared to the FP-Growth algorithm. N Painting-Growth algorithm and Painting-Growth algorithm presented in [28] reduce the overhead of scanning a database multiple times. The use of two-item permutation sets in these algorithms gives several advantages: running faster, occupying less memory space, having low complexity, and being easy to maintain. Based on the survey presented in our paper, it's evident that pattern-based growth algorithms like FP-Growth and their refinements based on the FP-tree are more efficient than algorithms based on candidate generation such as Apriori.

Conflict of Interest

The authors declare no conflict of interest.

Acknowledgment

I am grateful to the University of Colombo School of Computing (UCSC), Sri Lanka for all the facilities provided for this research work.

References

[1] H. Mahgoub, "Mining association rules from unstructured documents'', International Journal of Applied Mathematics and Computer Sciences, **1** (4), 201-206, 2008, doi:10.5281/zenodo.1330457.

[2] H. Mahgoub, D. Rösner, N. Ismail, F. Turkey, "A text mining technique using association rules extraction", International Journal of Computer, Electrical, Automation, Control, and Information Engineering, **2**(6), 2044-2051, 2008.

[3] M. Kulkarni, S. Kulkarni, "Knowledge discovery in text mining using association rule extraction", International Journal of Computer Applications, **143(**12), 30–35, 2016, doi: 10.5120/ijca2016910144.

[4] A. A. Lopes, R. Pinho, F. V. Paulovich, R. Minghim, "Visual text mining using association rules", Comput. Graph., **31(**3), 316–326, 2007, doi: 10.1016/j.cag.2007.01.023.

[5] Y. Li, N. Zhong, "Rough association rule mining in text documents for acquiring web user information needs", in 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006 Main Conference Proceedings) (WI'06), Hong Kong, China, 226–232, 2006, doi: 10.1109/WI.2006.151.

[6] J. Manimaran, T. Velmurugan, "A survey of association rule mining in text applications", in 2013 IEEE International Conference on Computational Intelligence and Computing Research, Enathi, Tamilnadu, India, 1–5, 2013, doi: 10.1109/ICCIC.2013.6724258.

[7] M. Delgado, M. J. MartÍn-Bautista, D. Sánchez, J. M. Serrano, M. A. Vila, "Association rule extraction for text mining", In: Carbonell, J.G., Siekmann, J., Andreasen, T., Christiansen, H., Motro, A., Legind Larsen, H. (eds) Flexible Query Answering Systems - FQAS 2002, Lecture Notes in Computer Science(), Springer, Berlin, Heidelberg, **2522**, 154-162, 2002, doi: https://doi.org/10.1007/3-540-36109-X_12.

[8] C. Lucchese, S. Orlando, R. Perego, "Fast and memory-efficient mining of frequent closed itemsets", IEEE Trans. Knowl. Data Eng., **18**(1), 21–36, 2006, doi: 10.1109/TKDE.2006.10.

[9] C. M. Rahman, F. A. Sohel, P. Naushad, S. M. Kamruzzaman, "Text classification using the concept of association rule of data mining", in 2010 Proceedings of the International Conference on Information Technology, Kathmandu, Nepal, 234-241, 2010, doi: https://doi.org/10.48550/arXiv.1009.4582.

[10] R. Agrawal, R. Srikant, "Fast algorithms for mining association rules" , in Proceedings of the 20th International Conference of very Large Data Bases, VLDB, Santigo, Chile, 487-499, 1994.

[11] T. Ouypornkochagorn, "Mining rare association rules on Banpheo Hospital (Public Organization) via Apriori MSG-P algorithm", ECTI Transactions on computer and information Technology, **6**(2), 156–165, 1970, doi: 10.37936/ecti-cit.201262.54337.

[12] V. Chandola, V. Kumar, "Summarization - compressing data into an informative representation", in 2005 5th IEEE International Conference on Data Mining (ICDM'05), 1-8, 2005, doi: 10.1109/ICDM.2005.137.

[13] Ali Ikhwan, Milfa Yetri, Yohanni Syahra, Jufri Halim, Andysah, Putera Utama Siahaan, Solly Aryza, Yasmin Mohd Yacob, "A novelty of data mining for promoting education based on FP-Growth algorithm", in 2018 International Journal of Civil Engineering and Technology (IJCIET), **9**(7), 1660-1669, 2018.

[14] Wei Zhang, Hongzhi Liao, Na Zhao, "Research on the FP Growth algorithm about association rule mining", in 2008 International Seminar on Business and Information Management, Wuhan, 315–318, 2008, doi: 10.1109/ISBIM.2008.177.

[15] R. K. Soni, N. Gupta, A. Sinhal, "An FP-Growth approach to mining

association rules", International Journal of Computure Science and Mobile Computation, **2**(2), 1-5, 2013.

[16] Satya Wacana Christian University, I. S. Rajagukguk, S. Y. J. Prasetyo, I. Sembiring, "The analysis and implementation of algorithm of Frequent Pattern – Growth to support the promotion strategy in Victory University Sorong", International Journal of Comput. Sci. Eng., **4**(10), 24–31, 2017, doi: 10.14445/23488387/IJCSE-V4I10P106.

[17] J. Panjwani, "Application of FP Tree Growth algorithm in text mining", M.S. thesis, Department of Computer Science and Engineering, Jadavpur University, Kolkata, India, 2010.

[18] K. Gadia, K. Bhowmick, "Parallel text mining in multicore systems using FP-tree algorithm", in 2015 International Conference on Advanced Computing Technologies and Applications (ICACTA), Procedia Computer Science , **45**, 111–117, 2015, doi: 10.1016/j.procs.2015.03.100.

[19] A. Kaur1, G. Jagdev, "Analyzing working of FP-Growth algorithm for frequent pattern mining", International Journal of Research Studies in Computer Science and Engineering (IJRSCSE), **4**(4), 22-30, 2017, doi: http://dx.doi.org/10.20431/2349-4859.0404003.

[20] Zulham Zulham, Ibnu Rusydi, Ananda H Elyas, "Pattern analysis of health equipment procurement system using the FP-Growth algorithmin", in 2nd International Conference on Inovations in Social Sciences Education and Engineering (ICoISSEE), **1**(1), 2021.

[21] M. Hasler, R. Karpienko, "Visualizing association rules in hierarchical groups", Journal of Business Economics**, 87**(3), 317–335, 2017, doi: 10.1007/s11573-016-0822-8.

[22] M. Hasler, S. Chelluboina, "Visualizing association rules in hierarchical groups", in 42nd Symposium on the Interface: Statistical, Machine Learning, and Visualization Algorithms (Interface 2011), Intelligent Data Analysis group, Southern Methodist University, 2011.

[23] O. Netzer, R. Feldman, J. Goldenberg, M. Fresko, "Mine your own business: market-structure surveillance through text mining", Marketing Science, **31**(3), 521–543, 2012, doi: 10.1287/mksc.1120.0713.

[24] J. Arora, "An efficient ARM technique for information retrieval in data mining", International Journal of Engineering Research & Technology (IJERT), **2**(10), 1079-1084, 2013, doi: 10.17577/IJERTV2IS100229.

[25] A. Bala, M. Z. Shuaibu, "Performance analysis of Apriori and FP-Growth algorithms (Association Rule Mining)", International Journal of Computer Technology & Applications, **7**(2), 279-293, 2016.

[26] C. Gy, "A comparative study of association rules mining algorithms", in 2004 1st Romanian- Hungarian Joint Symposium on Applied Computational Intelligence (SACI), 1-10, 2004, doi:10.13140/2.1.1450.3365.

[27] G. Alaghband, L. Vu, "A fast algorithm combining FP-tree and TID-list for frequent pattern mining", in 2011 Proceedings of Information and Knowledge Engineering, 472-477, 2011.

[28] M. N. M. García, S. Segrera, V. F. López, "Association rules: problems, solutions, and new applications", in 2005 Proceedings of the III National Workshop on Data Mining and Learning, 317-323, 2005.

[29] Y. Zeng, S. Yin, J. Liu, M. Zhang, "Research of improved FP-Growth algorithm in association rules mining", Scientific Programming, **2015**, 1–6, 2015, doi: 10.1155/2015/910281.

[30] I. Aqra, N. Abdul Ghani, C. Maple, J. Machado, N. Sohrabi Safa, "Incremental algorithm for association rule mining under dynamic threshold", Applied Sciences, **9**(24), 5398, 2019, doi: 10.3390/app9245398.

[31] M. Hasler, "arulesViz: interactive visualization of association rules with R", The R Journal, **9**(2), 163-175, 2017, doi: 10.32614/RJ-2017-047.