National College of Ireland

Using the top-k frequent item set for mining non-overlapping patterns

MSc Research Project
Data Analytics

Shivam Gulve
Student ID: x20181400

School of Computing
National College of Ireland

Supervisor: Dr. Hicham Rafai

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Shivam Gulve
Student ID:	x20181400
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Hicham Rafai
Submission Due Date:	15/12/2022
Project Title:	Using the top-k frequent item set for mining non-overlapping patterns
Word Count:	8159
Page Count:	23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Shivam Gulve
Date:	30th January 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Using the top-k frequent item set for mining non-overlapping patterns

Shivam Gulve
x20181400

Abstract

Identifying frequent patterns where sub patterns are not repeated is called non-overlapping maximum sequence pattern (NMSP) mining. Applying non-overlapping pattern discovery mining, a unique kind of repeating sequential pattern mining which uses gap restrictions, it's indeed possible to discover more useful patterns. Recent techniques focused on discovering common patterns, which resulted in the identification of more compact, repeated patterns. However, it also makes it much more challenging to attempt to identify, necessary information, which has an impact on the effectiveness of mining. We have put forward the efficient modified mining algorithm NetNMSP and will give a set of data, and then in order to identify unique NMSP patterns, it will provide the pattern count to create a sequence. The next step would be to provide top k count i.e., pattern count and pattern size. Basically, there will be a pattern count present and pattern size, so it will look for a range of patterns. Applying the pattern joining approach, which initially apply the joining method to create common patterns, is the third phase. So, at the end, then there will have a candidate pattern for maximal sequential patterns that do not overlap. The proposed system have remove minimum support dependency and this system is efficient based on processing time and processing memory.

1 Introduction

In a world shaped by data, data mining play a major role in identifying knowledge within data, Sometimes referred as knowledge discovery from data. Data mining is interdisciplinary and includes a variety of methods. It encompasses a broad range of topics, including prediction, classification,pattern mining. Pattern mining has been demonstrated to be important and it include several methods. The simplest consists of finding the most frequent items in a dataset for example, finding the most frequent item sold in a shop. Sequential pattern identifies the most frequent sequence of items.There are several application that uses sequential pattern mining such as identifying the buying patterns of customers, that is the sequence of products bought by a customer. Identifying browsing behaviour of the user, forcasting natural disaster based on a sequence of eventLi et al. (2022).

In contrast to traditional sequential pattern mining, recurring sequential pattern mining takes into consideration whether frequently the pattern appears in such a sequence. So, mining repeated sequential patterns could provide additional data. Finding frequent

patterns where sub-patterns are uncommon is the goal in non-overlapping maximum sequence pattern (NMSP) mining.

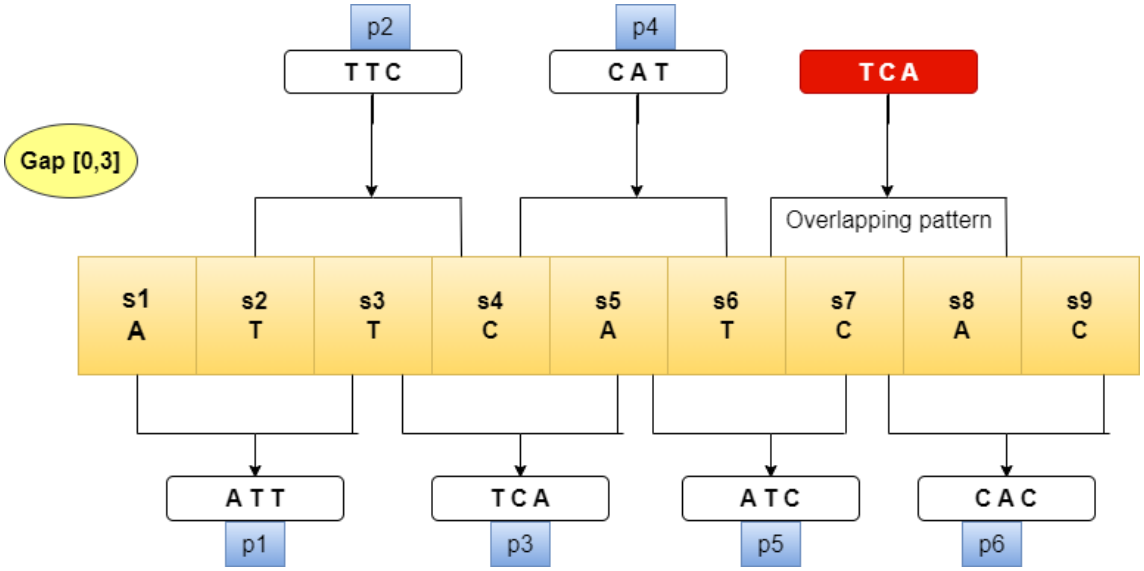

Figure 1: All-type of repetitions of pattern P in sequence S

A non-overlapping minimum-support, a significant problem in non-overlapping sequential pattern mining, is shown in Example 1 Wu et al. (2017).

Example 1 Let us assume there is an sequence S1 = ATTCATCAC and a patterns P= 'A','C','T' with gap= [0,3]). All types of occurrences are shown in Fig 1

Example 1 we can see that A type of pattern having a gap limitation is pattern P and which also have 6 patterns occurrences in sequence-S1. We have sequence s = ATTCATCAC= s1, s2, s3, s4, s5, s6, s7, s8, s9 with constraint gap of [0,3]. So, to begin with given sequence we see 'A','T', 'T' so we get our fist pattern p1 = ATT. As, we found our fist pattern we go for next sequence s2 we get p2 = TTC so on p3 = TCA, p4 = CAT, p5 = ATC, now we move on for sequence s6 we get 'T','C','A' but if we see pattern p3 we already got same pattern. So, this is overlapping pattern but we have gap of [0,3] then we can move to next sequence s7 i.e., 'C'. So, we get our pattern p6 = CAC. After finishing our iteration, we found 7 frequent patterns ATT, TTC, TCA, CAT, ATC, CAC. The predefined repetition tree technique's basic idea is to add a character to a end of the each existing pattern and develop new patterns. Each pattern can thus produce candidate patterns for NMSP.

Recent studies suggested that the execution of mining problem with both the Apriori and Netback method seems to be suitable for non-overlapping sequential pattern mining. To compared with those other new techniques, it's indeed simpler and can be useful for common patterns that resolves issue related to patterns problems. The non-overlapping minimum support, is a major problem with non-overlapping sequential pattern mining. Regardless of whether the minimum support value was chosen it could be difficult for a data analyst as it define a suitable limit. For most cases, it's indeed inefficient to run such type of algorithm which gradually increases the value. If the minimum support value is

selected wrong, any algorithm may be unable to understand the real patterns.
Matching mining was used to identify statistically significant patterns. It could be challenging for a data analyst that set a suitable predetermined threshold, even if the minimum support value is chosen somewhat randomly. In the majority of circumstances, it is necessary to repeat this procedure continuously until the required result is obtained, constantly decreasing the value for minimal support across such a broad range. The algorithm might be unable to identify actual true patterns if indeed the minimum support value was chosen improperly. These offer a special approach to immediately retrieve the top few highly popular patterns by priority order while requiring a minimum support value. Non-overlapping sequential pattern mining, a modification on recurring sequential pattern mining with gap constraints, enables the discovery of more valuable patterns. Existing techniques concentrated on discovering common patterns and found several short, repetitive sequences as a result. Non-overlapping maximum sequence pattern (NMSP) mining is the process of identifying consistently occurring patterns and uncommon subpatterns.

The primary objective of this project would be to utilize a large number of patterns for sequence pattern mining, which is frequently used in so many different fields. However, most of these patterns are significantly more challenging to utilize in practical situations. Therefore, identifying the more frequent k pattern was necessary to find the top-k pattern, and this is recognized as just an effective method. Knowing that pattern which appears most frequently is among the most important pieces of data for a user. Furthermore, we get the option of decreasing the minimal support value.

1.1 Research Question

What is the ideal strategy to efficiently determine the non-overlapping patterns from a set of sequential patterns ?

2 Literature Review

2.1 Background information on types of pattern mining

Finding unexpected as well as useful patterns across databases represents one of the main goals for data mining. Creating algorithms which extract patterns using sequential data appears to already have gained prominence in recent years across the field of data mining. In data mining, one of the most commonly used methods analyzing sequences is sequential pattern mining. It requires locating fascinating sub sequences within such a collection or sequences, where the interestingly of the a sub sequence may well be assessed using a variety of elements, such as its length and frequency of appearance. Sequential pattern mining has a broad range of practical applications since information is frequently represented in sequences in disciplines including informatics, e-learning, market basket analysis, text analysis, and web page click-stream analysis. This describes recent work into sequential pattern mining and applications. The goal is to present a summary of recent advancements, research opportunities, and is something akin to the an introduction to sequential pattern mining. The initial step is to determine and assess sequential pattern mining's function in addition to its applications. The key strategies and approaches

towards resolving problems involving sequential pattern mining also are covered. Several iterations of both the sequential pattern mining method are discussed, as well as highlighted how limited traditional sequential pattern mining methods are. It highlights potential research possibilities in addition to how closely related the issue is to including well pattern mining problems. Accessible applications using sequential pattern mining algorithms also are addressed in thisFournier-Viger et al. (2017).

This offers up-to-date review for sequential pattern mining as both a method for resolving fundamental issues. It really can serve both as an introduction towards the field as well as a resource regarding current research and job opportunities. The remaining sections of the that type of paper are organized as follows. That this next chapter explains fundamental issue of sequential pattern mining in addition to the main techniques employed all throughout process. The following section addresses typical generalizations of both the sequential pattern mining problem along with a few additional data mining problems that appear to still be closely related with sequential pattern mining. The report then examines prospective avenues for research in addition to free open source implementation for sequential pattern mining techniques. However, due handle the enormous amount of information that needs to be analyzed within real time, advanced computer systems are necessary. High performance computing (HPC) systems offer robust computation that makes it possible to process huge data effectively. Creating algorithms for data mining that can deal with a wide range forms data abilities and uncover hidden relationships and different patterns behind them is a hard topic of data scientists. When data increasing capacity, existing algorithms, instruments, and processes must operate in real time within such a special computer environment. Data mining's fundamental subject in sequential pattern mining assists in there anyway examination of so many critical facts regarding identifying sequence patterns.

Following are some of the other most effective methods of spotting sequential patterns which use the prefix span. Additionally, but since sequential program for prefix span uses what amounts to something like a single CPU, this takes awhile to calculate any sizable sequence database. Inside an HPC system, different processors interconnected through a high-speed connection can perform the same calculation simultaneously. The goal would be to employ a heterogeneous computing system to shorten overall overall execution time for prefix spans.

2.2 GPU Computing Mining

They suggested a technique which utilizes the NVIDIA CUDA platform to implement these two basic prefix span jobs concurrently. The two primary objectives are also to identify often occurring things and also to develop projected databases. This study presents a novel approach for creating projected databases in which only the beginning and end indexes for a particular series may well be retained in the some type of the a single array. The memory consumption of the HPC system could be decreased as a result. The creators of HPC systems have such a large number of processing cores Zhou et al. (2010). High-speed PCI bus links that connect the CPU and GPU could be combined alongside heterogeneous architecture for high performance computers (HPCs) Wang et al. (2013). The CPU and GPU can in fact compute identical instructions at the same time.

The adoption using parallel processing methods could lead to extraordinarily powerful computers. Since so many processes may well be executed in parallel using the graphics processing unit (GPU), parallel processing using GPU can reduce execution time Karthik and Saira Banu (2020). NVIDIA became the first company to introduce GPGPU, meaning general-purpose computing on the graphics processing unit. GPU has indeed been utilized both to create and compute visuals Jiang and Meng (2017). Phan (2019). It is significantly efficient than just a CPU at performing a broad range of tasks simultaneously, particularly complex mathematical processes.

Pattern mining has indeed been extensively used across an wide range of fields. Numerous patterns are regularly extracted by users. The majority of the these, however, were difficult to utilize in real-world scenarios. The method of finding the top-k patterns is known as "top-k pattern mining," and it is old since a pattern seems to be more likely to matter to users if something occurs frequently. However, for mining applications that exploit on Apriori property, top-k mining could only efficiently mine short sequences. Smaller patterns have been shown to have much lower information density than longer patterns. The specialize in mining the top k sequence patterns for every pattern length in this section. The technique examines the Nettree data structure to see whether a consistent pattern was accepted. The data structure was developed in order to address various pattern finding and sequence pattern mining challenges. To generate the new top k super-patterns using len + 1, researchers first discover a most important k patterns having length len, next calculate overall supports of everything matching the related k sub-pattern with length-len + 1. The algorithm outperforms comparable approaches, according to experimental results. These are the three main insights drawn from the study:
(1) Since technique NOSTOPK does not call again for specification of the minimum support threshold, it solves the problem of how difficult it can be to specify the minimum-threshold for frequent pattern mining.
(2) Top-k patterns among all common patterns are mined by conventional top-k algorithms. The issue of mining the top k things for every length is something they take on. Users additionally select the required pattern length L and the required number generated patterns k.
(3) They illustrate their strategy's practicality and effectiveness that use the effective method NOSTOPK and significant experimental results Chai et al. (2018). The method that's also widely used to use for data mining was cluster analysis mining. This method may uncover unexpected connections among groups of objects inside a data set such as this and predict correlated activity with new data. It needs to be decided in advance whether specific items will be part of or referred as the frequent item set before such a rule is established. Objects left out from either the frequent item set or even the minimum support were eliminated to use a threshold in the this step.
The criterion does have a significant bearing on how many rules are generated as well. Nevertheless, association rule mining won't being able to find anymore rules if indeed the threshold is picked wrongly. The current minimum-support-value was selected at random either by user. This presents an issue that is more difficult for just a user who is really not conversant with features of a dataset. Time and memory are also both intensively utilized. The reason for this is because the regulation process gets iterated until the ideal amount of rules is identified. Every transaction's average average overall item count, in addition to the worth of the each item's support. The total average number of items included in each transaction is utilized to calculate the minimal support value as

a component of a flexible recovery program, as well as the appropriate support values. The suggested technique additionally employs predetermined threshold factors, therefore the final regulations continue to remain line with the user's preferences. The proposed methodology establishes an ideal minimal support value while taking into account both the total number all engaged thoughts as well as the average utility.The experiments were conducted using U separate datasets and determine the association rules employing different dataset properties. The mainly two association rule algorithms employed in the examination of the proposed adaptive support method were apriori and cpgrowth. The test were repeatedly run to determine the highest and lowest minimum support values. The results showed that now the minimum and maximum support values for S from our U datasets now are provided by the apriori and fpgrowth methods. Whenever analyzed from either the quality that it creates at a lift ratio value significantly larger then 1, the data from both the recommended adaptive support does possess the ability to create a rules. When determining this minimal threshold value, it could be appropriate for take into consideration certain characteristics of a dataset that have been identified thru the experimental findings Hikmawati et al. (2021).

2.3 User based Minimum Support

The user first should determine a minimum support value as a part of the fundamental design of something such as the association rule. The value can frequently applicable to every things, even though different products may well have distinct assessment methods. As just a result, numerous minimal support studies have been conducted out, and these demonstrate that according to the item, the average value of both the support should differ Dahbi et al. (2017). Due to implementation of the system, a user has now extra work to undertake, specifically determining the bare minimum degree of support for each item. A new line of research into frequent patterns without minimum support also has been opened up as a consequence of both the difficulty in establishing user-determined minimum support levels, such Top-K Ryang and Yun (2015) and Skyline. Since the user was essentially needed to choose an of k, that reflects the number of rules that would be generated during in the rule creation, the Top-k association rule actually did not require a minimum support value in the this technique. The fact that users all are aware of the number of rule possibilities they desire makes is easier to them to all determine that k value.

For somebody, one of the challenging components about applying frequent patterns was selecting a minimum support value. This is due to practically all methods work under the presumption the database objects were nearly always equal or regularly occurring. This assumption is incorrect though, like some things can be discovered in the database much more frequently than others Trivedi and Patel (2017). Because existing algorithms such apriori or fpgrowth could indeed determine the minimal support and threshold values, and user makes these assumptions based on their intuition. The mining of association rules technique can generate a lot more rules, which could cause this to take a very long time to execute and consume as large amount of memory. However, everything here is dependent on just that threshold choice. Because users experience problems deciding on such a minimum degree of support, Apriori-based mining methods were created and have now become widely used item sets. When attempting to increase efficiency, the approach

creates a difficult paradox because it becomes so reliant on the a user-defined threshold. If the minimum support value is set to the an overly high value within this situation, the database gets cleared. On the contrary hand, low minimum support has a number of unfavorable regular patterns, including such ineffective mining. As just a result of either one of those two reasons, users already are unfairly required to identify the specifics of the database that must also be mined in addition to an appropriate threshold. Even when professional miners oversaw the investigation of both the minimum help, neither of the results even came close to meeting that user's expectations Zhang et al. (2008) .

By including the average support threshold, Trivedi Trivedi and Patel (2017) studied the Semi-Apriori technique. Common items were again evaluated to ascertain relevant data using an autonomously formed resource dependence which constructed frequent item sets more frequently. Above there decreases the challenge in both space and time.

Dahbi came up with a method for selecting a suitable minimum stated level that would provide aid Dahbi et al. (2017). This experiment continually produced optimal minimum support (minsup) with each data set instead of using a user-defined reference value. The second dynamically updated the minsup because it finished via using a single, uniform minimum support threshold to each and every level. However, not that every part of the a set works in the exact same way; some parts are being used frequently while others are only seldom. As just a result, the minsup threshold must adjust based on the item level.

2.4 Different approach for Pattern Mining

Analytic are essential for the process of making important decisions. The advantages of these pattern analysis insights are enormous, among which are enhanced profitability, reduced costs, more competitive advantage. However, as even the amount of information increases over time, it takes longer effectively mine the underlying patterns of something like the common item sets. Because to such algorithm's intensive computation, significant memory consumption is also required for mining underlying underlying patterns in frequent item-sets. Consequently, whereas the amount of data grows over time, a effective method is needed to mining underlying hidden patterns of the frequent item-sets in a faster run time and with lower memory requirements. Throughout order to create a more effective FPMs algorithm, research study analyses and compares numerous Frequent-Pattern-Mining (FPM) algorithms. Each business must analyze all of the data that have been collected in the data store or warehouse in order to make good decisions by taking into account all the data sets. It is crucial to mine to find all the key underlying patterns that regularly occur in a data in order to give consumers information that is more valuable for data analysis and decision-making. To give an overview of the state-of-the-art in FPM, the paper analyzes a variety different FPM algorithms. the earlier research with FPM algorithms. While there is a table which contrasts the basic and important FPM algorithms that have recently been put out by different academics Chee et al. (2019).

Data mining is a method that examines huge amounts of information to identify the most important patterns. The paper examines a few data mining methods, algorithms, and enterprises that have used data mining technologies to improve business operations

with great success. Like a outcome of studies in databases and information technology, one method of storing and manipulating this precious data for the future decision-making has developed. Data mining is the process for sifting over huge amounts of data to discover pertinent patterns and data. Other names for just that include pattern analysis, information mining using data, knowledge extraction, and information gathering method. Data-mining is a method that used collect useful data from the a huge amount of data. This approach aims to discover previously unknown patterns. These patterns will then be utilized as direct particular industry suggestions to management after they have been discovered Bharati and Ramageri (2010).

Data mining approaches have indeed been intensively developed by researchers. That entails investigating novel forms of knowledge, mining in multidimensional space, combining techniques from various fields, and taking the relationships between data objects' semantics into account. By addition, concerns like data uncertainty, noise, and incompleteness should be taken into account by mining algorithms. Several mining techniques investigate the use of user-specified metrics both evaluate the novelty of data patterns and to direct pattern discovery. Let us just examine these many facets the mining techniqueJiang and Meng (2017).

2.5 Conclusion

To conclude our literature review we can say that all different types of paper shows different approaches of finding the non-overlapping maximal sequential patterns. They use different types of algorithm like apriori, netback, netnmsp algorithm and much more. After implementing successful algorithm they conducted different types of experiments and came with different results.

3 Methodology

To identify the common pattern was an objective of non-overlapping sequential pattern mining. The size of mining pattern collection and availability duplicate short patterns are some of its disadvantages. It is recommended that the problem of removing duplicate patterns should be solved by using non-overlapping closed sequential pattern mining, that essentially states there was not a same pattern that shares the same support. The approach we will be following is KDD Data Mining methodology and the method we are using is by proving minimum and maximum gap between finding patterns. Using sub-sequences that have a larger gap compared to those with smaller gaps, the method performed well for this type of collection of data. Some method of eliminating duplicated patterns is called Non-overlapping maximal sequential pattern mining (NMSP), that makes sub-patterns with patterns that matches absolute rare patterns. The efficiency of maximum pattern mining is better than limited pattern mining.

Calculating minimum support, reducing potential patterns, and recognizing NMSPs have been the three main variables affecting the mining effectiveness when dealing with NMSP mining. Again for Netback technique that ascertain a pattern's support within such a Nettree, a backtracking mechanism is necessary. utilizes using pattern join method to create probable patterns.

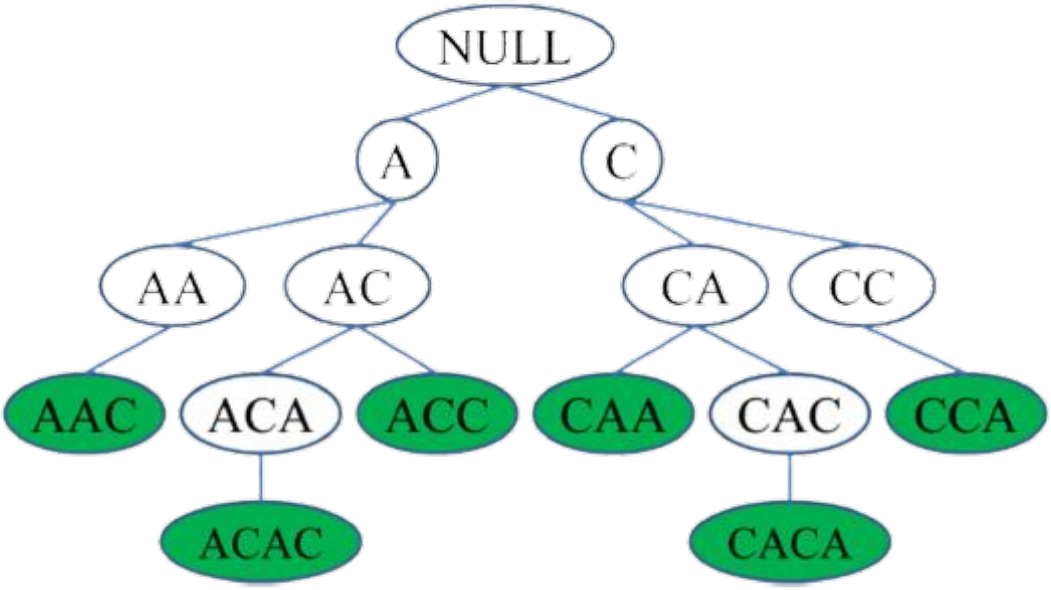

Figure 2: Correlation of NMSPs with common patterns

The process of discovering frequent sub-sequences is referred as a patterns that are not required to be in continuous also shown in 2. It is possible to identify dependencies utilizing repeating sequential pattern mining without even any gaps limitations. A method of sequential pattern mining without gap limitations is called non-overlapping sequential pattern mining (or sequential pattern mining).When we take into consideration the non-overlapping constraint, every item in this sequence could only match another time.

3.1 Nettree and Algorithm Design

As we discussed about different types of pattern mining problems and to resolve it we need an efficient algorithm which can identify the non-overlapping patterns. Since a node could have more than a parent-node, using Nettree-algorithm, which is essentially a distinct type of tree, will be utilized in this.The name Nettree for pattern finding is basically based on Nettree which is designed to solve pattern problem. The Nettree would be a type of DAG(Directed-Acyclic-Graph) with two types of node names, "parent-child" or "child-parent," in which all nodes contain zero or more children-nodes and zero or more parent-nodes. Therefore, the children of-each nodes have a different-ordersWu et al. (2010).

There are three different kinds of attributes in a Nettree algorithm:

1. Since it includes the root, leaf, level, parent, child, and so on a nettree is indeed an extended form of the a tree - like structure.

2. When r $>$ 1, a Nettree can have r roots.

3. Inside a Nettree, certain nodes beyond the roots could have q parents, with q1$\geq$1.

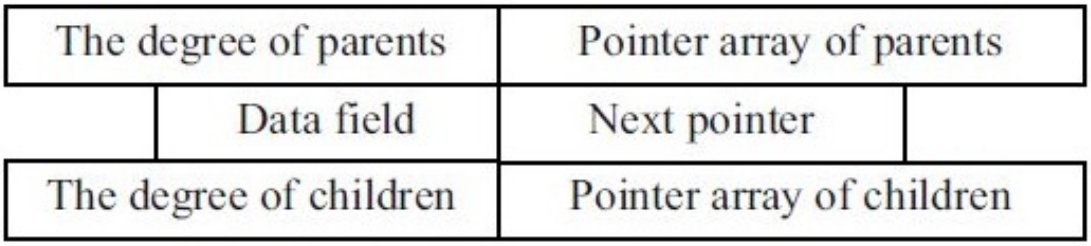

Figure 3: Nettree Structure

All node types are connected to use the Nettree, as well as the number of nodes at each level was counted. Again for Netback technique can ascertain a pattern's support within a Nettree, a backtracking mechanism is necessary. There are two sections throughout this Nettree structure: data as well as pointer. Character and integer data types, that indicate the similar characters of each node and the number the nodes in the this level, correspondingly, are both present in a data section. There are three pointers inside a pointer section: the header pointer, start pointer, the tail pointer. The very first node in the this level is to which the head-pointer was pointing. The last pointer, sometimes known as that of the tail-pointer, points towards the final node of the this level whereas the start-pointer points to the first parent of a node on the following level.

As shown previously, there have been six fields inside the construction of the Nettree nodes. The numbers of the its parents and children were represented, respectively, either by degrees of parents and children. The pointer arrays containing parents and children, respectively, represent both parents and children of a head node. The very next pointer points to a node which will replace the existing node with the same level. Two integers are contained in a data field, one for the sequence's location and the other for the number of its root pathsWu et al. (2010).

A Nettree flowchart is shown in fig4. The Nettree's two main roots were Nodes A and B. Three branches contain nodes like C, F, and G. Having just two parents, Node D. (nodes A and B). Every edge in Fig. is identified either "parent-child" and "child-parent" by a named label. As a result, it's a DAG containing edge labels. Meanwhile, a cycle of "B, D, G, E, B" would occur.

A directed-acyclic-graph may logically describe a collection of actions (DAG). The order of a activities can be seen using a graph, which would be graphically depicted as just a collection of circles, a few of which are connected with lines to indicate the flow through one action to another.

3.2 NetNMSP Architecture

As we are discussing about a type of methodology we will be using to implementation, above we can see the system architecture and it has mainly classified into seven stages. First stage is to provide a top k count for sequence. We will having a set of data and to figure out the NMSP patterns we have to provide the number of count for creating sequence. Then second step is to provide the minimum support and find the top k patterns. In this we will basically provide a support value and it will find number of patterns. The third step will be to apply the pattern joining strategy, that is to apply

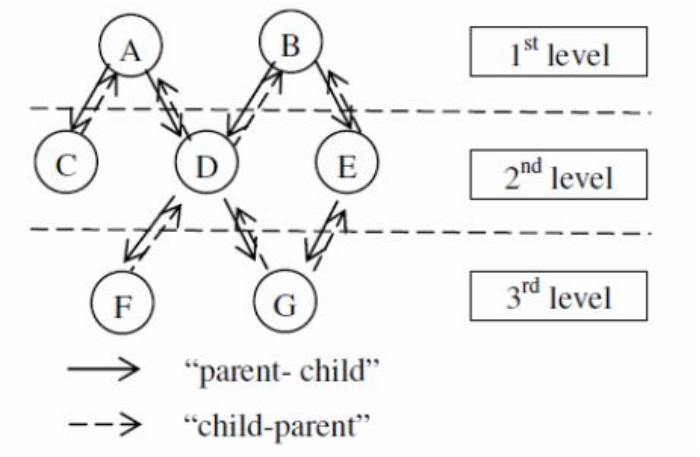

Figure 4: Flow of Nettree

joining method and it will create common patterns which we showed in fig.2. Then all types of common patterns are found we will be finding valid candidate for NMSP (Non-overlapping Maximal Sequential Patterns). After finding the candidate for NMSP we will be getting our non overlapping maximal sequential patterns.

3.3 Data Set

We will be evaluating the performance of an enhanced NetNMSP algorithm applying experimental data using DNA, protein, and viral sequences.

1. The DNA sequence from Homo-Sapiens Al-158070 can be downloaded from `https://www.ncbi.nlm.nih.gov/nuccore/AL158070.11`
2. Protein sequences could be found inside the ASTRAL database, which can be download from `https://scop.berkeley.edu/astral/subsets/ver=1.61`
3. Baby-sale is a sales dataset for baby products which can be obtained via `https://tianchi.aliyun.com/dataset/45`
4. Superstore is a sales dataset that can be downloaded from Super-Store link `https://tianchi.aliyun.com/dataset/93285`.
5. 200 human genes, both positively and negatively, are located at TSS (Transcriptional Start Sites), that comes via `https://dbtss.hgc.jp/`
6. The gene sequence of virus that created SARS-CoV-1 (Severe Acute Respiratory Syndrome Coronavirus 1) early 2003 were reported by Reference He et al. (2004) and can be downloaded at `https://www.ncbi.nlm.nih.gov/nuccore/30271926?report=fasta`
7. The genome sequence of a viruses that cause COVID-19, called as SARS-CoV-2 (Severe Acute Respiratory Syndrome Coronavirus 2), and can be downloaded from `https://www.ncbi.nlm.nih.gov/nuccore/MN908947.3?report=fasta`
8. Downloading MERS-CoV (Middle East Respiratory Syndrome Coronavirus) is possible via `https://www.ncbi.nlm.nih.gov/nuccore/NC_019843.3?report=fasta`

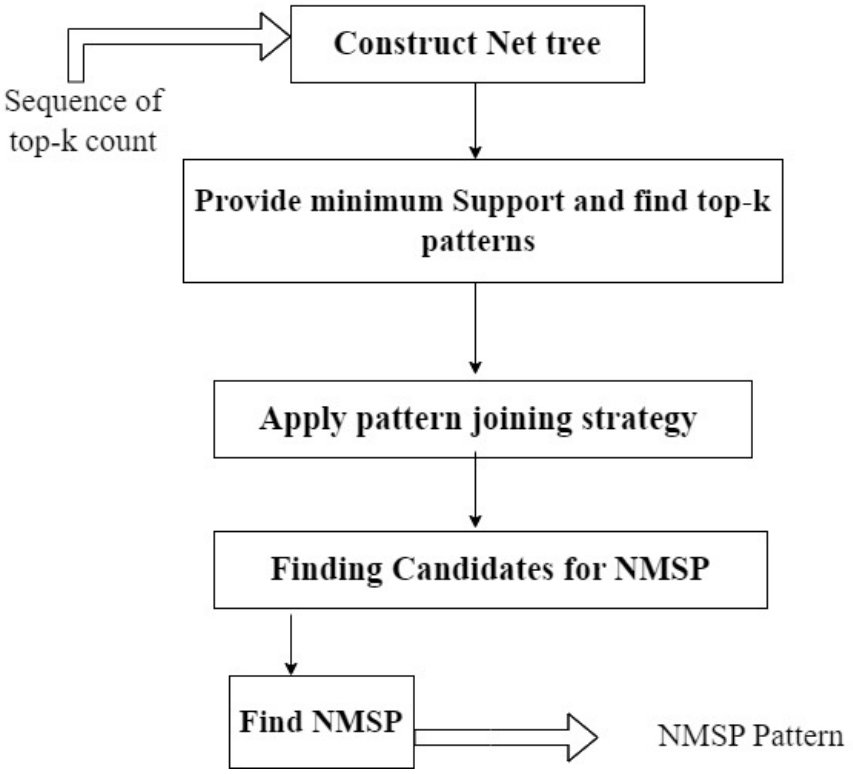

Figure 5: Architecture of NetNMSP

The data set for this project we will be using for evaluating different type of result can be download from here `https://github.com/shivamgulve4/research-project-data-set-/raw/main/DataSet.zip` this link. The data set contain different type of string patterns and each pattern has there own definition, like DNA data set has DNA patterns, virus has different types of string pattern and so on. There is no null or any garbage value present in data set, so we did not have to perform any pre-processing on data set.

4 Design Specification

4.1 NetNMSP Algorithm

TSupport threshold and non-threshold based algorithms is two distinct types of Top-k mining algorithms. The suitable minimum support threshold parameter was required from one of the most recent algorithms that also are frequently utilized during pattern mining methods. The existing algorithm works on finding NMSP's candidate patterns with minimum support. However, we have already talk about, it is very inefficient in practice to conclude the accurate value for this parameter As, an outcome, a new category for threshold-free Top-k frequent item sets algorithms is developed. So, in our algorithm we will be providing pattern size and minimum/maximum gap for finding sequential patterns. We have also provided pattern count which will be helping us to get sufficient number of patterns. So, it also improves the efficiency of algorithm.

Dataset	Type	Source	Number of sequences	Description	Total length
DNA1[1]	DNA	Homo Sapiens AL158070	1	Single	6,000
DNA2	DNA	Homo Sapiens AL158070	1	Single sequence	8,000
DNA3	DNA	Homo Sapiens AL158070	1	Single sequence	10,000
DNA4	DNA	Homo Sapiens AL158070	1	Single sequence	12,000
DNA5	DNA	Homo Sapiens AL158070	1	Single sequence	14,000
DNA6	DNA	Homo Sapiens AL158070	1	Single sequence	16,000
SDB1[2]	Protein	ASTRAL95_1_161	507	Multiple/unequal length	91,875
SDB2	Protein	ASTRAL95_1_161	338	Multiple/unequal length	62,985
SDB3	Protein	ASTRAL95_1_161	169	Multiple/unequal length	32,503
SDB4	Protein	ASTRAL95_1_171	590	Multiple/unequal length	109,424
SDB5	Protein	ASTRAL95_1_171	400	Multiple/unequal length	73,425
SDB6	Protein	ASTRAL95_1_171	200	Multiple/unequal length	37,327
Baby1[3]	Babysale	Sales of baby products	1,636	Multiple/unequal length	73,272
Baby2	Babysale	Sales of baby products	2,077	Multiple/unequal length	94,152
Baby3	Babysale	Sales of baby products	2,544	Multiple/unequal length	115,088
Baby4	Babysale	Sales of baby products	3,057	Multiple/unequal length	137,941
Super1[4]	Superstore	Superstore time series	1	Single sequence	100,001
Super2	Superstore	Superstore time series	1	Single sequence	120,001
Super3	Superstore	Superstore time series	1	Single sequence	140,001
Super4	Superstore	Superstore time series	1	Single sequence	161,048
TSS[5]	Human genes	Transcriptional Start Sites	200	Multiple / equal length	20,000
SARS-CoV-1[6]	DNA of the virus	Severe Acute Respiratory Syndrome Coronavirus	1	Single sequence	29,751
SARS-CoV-2[7]	DNA of the virus	Severe Acute Respiratory Syndrome Coronavirus 2	1	Single sequence	29,903
MERS-CoV[8]	DNA of the virus	Middle East Respiratory Syndrome Coronavirus	1	Single sequence	30,119

Figure 6: Description of Data Set

4.2 Proposed Algorithm

The below pseudo code shown in fig7 is our proposed algorithm and it shows how our algorithm will be working. So, to begin its explanation with working first step is to required pattern size or top k (i.e., minimum support) then it will scan sequence for data set or either we can provide input i.e., pattern sequence. The following phase will involve create the NetNMSP candidate implementations C with the a pattern length of m + 1. This will done using the frequent pattern set Fm as well as the pattern size. The minimum support for pattern P in set C would then be estimated using NetNMSP. We will check for each sequence S in data set we will add support value with sequence, pattern and minimum gap. Now, we will check the condition if top k is less that top k size and if yes then we will store the frequent patterns in Fm + 1. Now, we will delete prefix and suffix of pattern p in frequent pattern Fm. Then we check if minimum gap is greater than support then store support in minimum gap. Now, we will check if min is less that support then we will remove the Fm + 1 and min value. Prefix and suffix patterns in Fm then will be eliminated as they are not NMSPs. The program would keep repeating the procedure indicated below until enough C patterns have indeed been examined. The remaining patterns of Fm are NMSPs and thus are kept in Fmax. The technique described above would be repeated till pattern set C becomes empty by NetNMSP. All extra patterns of Fm have NMSPs and are therefore kept in Fmax. The procedure outlined above was carried out till NetNMSP determines pattern set C to still be blank.

Algorithm Min all NMSPs.

```
Require: Pattern size / top k
1: Scan sequence for data set (DS)/user input and calculate the support of each
   patterns
2: m <-- 1;
3: c <-- PatternJoin (Fm, PatternSize);
4: while c≠NULL do
5:    for each p in c do
6:       support <-- 0;
7:       for each s in DS do
8:          support <-- support + Netback (s, p, gap);
9:       end for
10:     if top k < top k list.size()
11:        Fm + 1 <-- Fm +1 U  p;
12:        Delete Prefix(p) and Suffix(p) in Fm;
13:        if min > support
14:              min = support
15:        endif
16:     else
17:       if min < support
18:       Remove (Fm + 1, min);
19:       Fm + 1 <-- Fm + 1 U  p;
20:       Delete Prefix(p) and Suffix(p) in Fm;
21:       Find min from Fm + 1;
22:     end if
23:    end for
24:    Fmax <-- Fmax U Fm;
25:    c <-- PatternJoin (Fm + 1);
26:   end while
27:  return Fmax;
```

Figure 7: Proposed NetNMSP Algorithm

5 Implementation

The objective in this research aims to more identify relevant non-overlapping maximal sequential patterns from group of patterns. So, to achieve our objective we have developed an upgraded NetNMSP algorithm where the existing one was only use to have an minimum support or top k to find NMSP patterns but in our we have proposed an method of finding patterns through giving top k count (i.e., pattern count) and pattern size. So, we build an application for determining the NMSP patterns through multiple approach. To build this application we used Java version 15.0.1 and JDK 11. The user interface is build with help of Apache NetBeans 12.4 and for building the NMSP algorithm we have used java as our primary language.

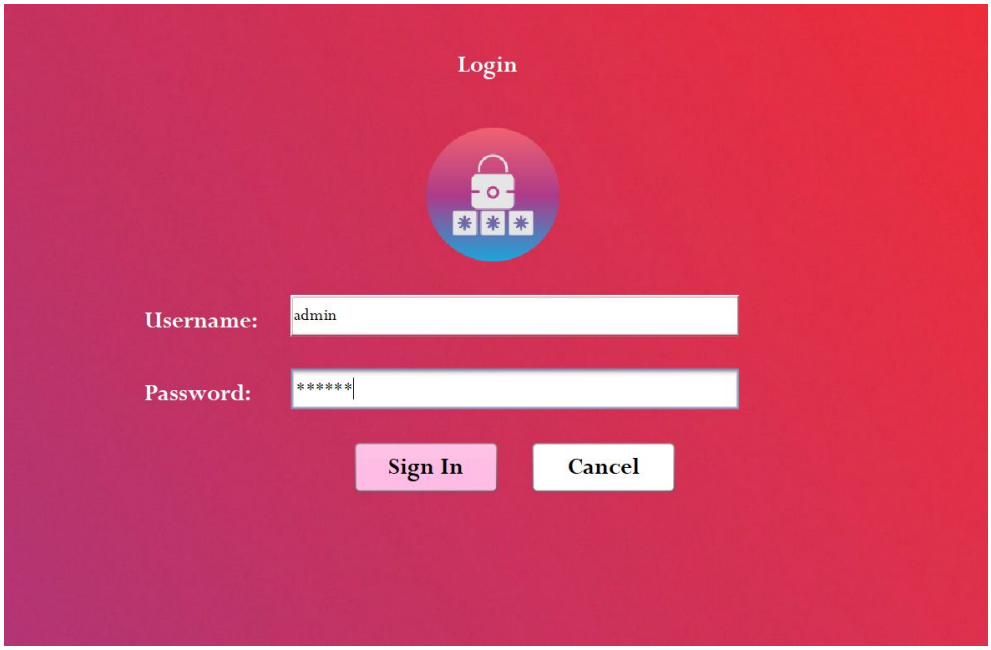

Figure 8: Login Page

After successful login and authentication, the control panel will be the next phase. The three choices are accessible through the control panel. The first choice is to find NMSP patterns using random patterns. Second choice is to find patterns using data sets or sing-out, as seen in fig9.

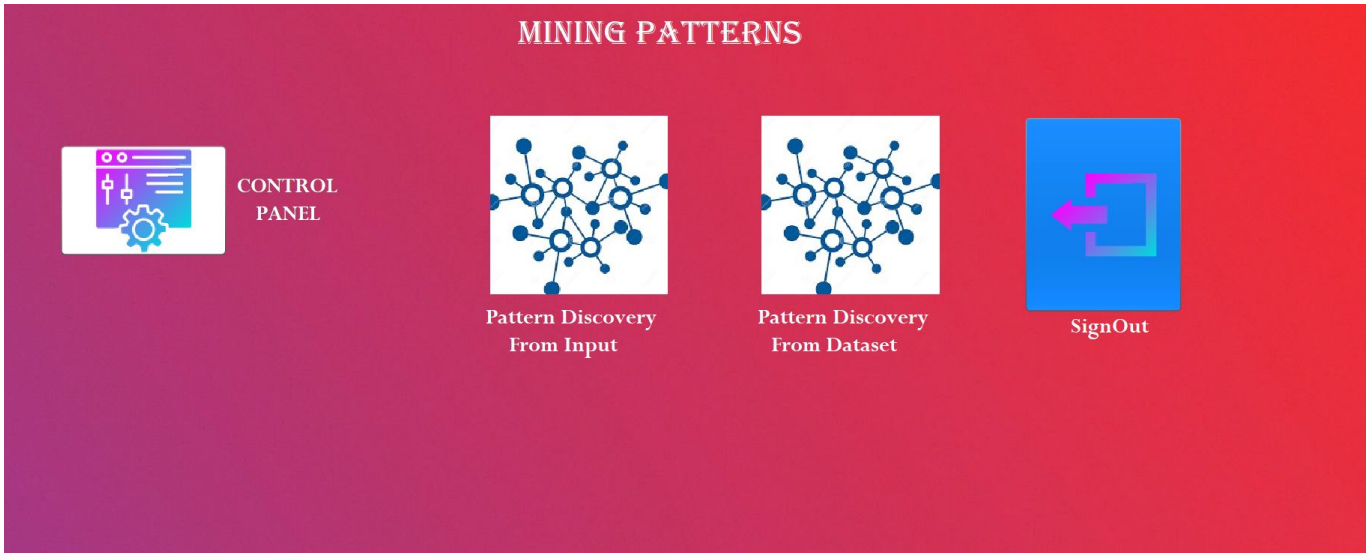

Figure 9: Control Panel

Let's assume that we have chosen the first choice, which is Pattern Discovery from Input shown in fig10 . In order to achieve non-overlapping patterns, we must first add data, i.e., the pattern we wish to examine. After submitting a random pattern, we must determine the minimum and maximum gap before choosing the method to be applied. Either the minimum support technique or the pattern count and size are options (i.e., proposed method). After providing all necessary information, we must submit all the details and wait for the results.

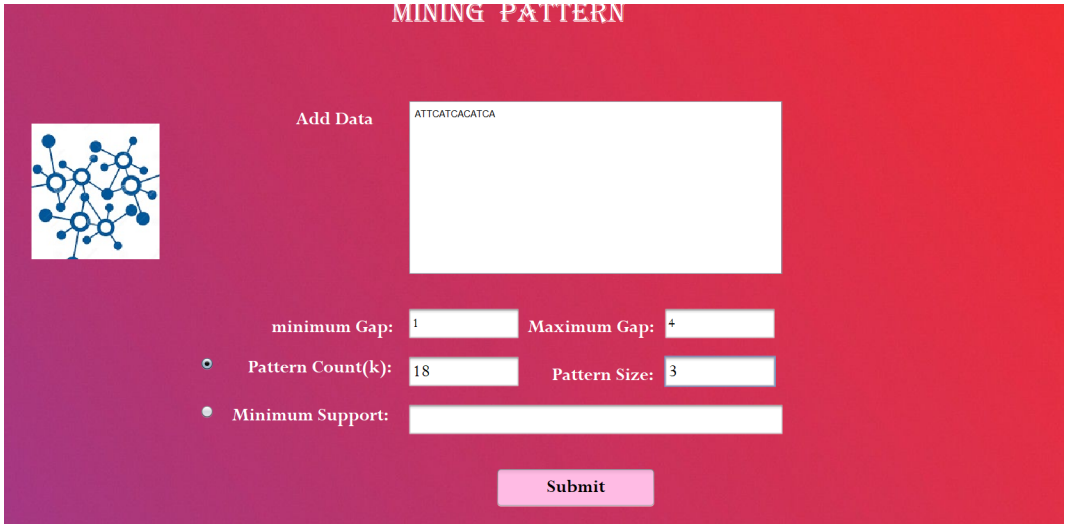

Figure 10: Pattern Discovery from Input

So, to see the demo result for Pattern Discovery from Input is shown in fig 10 and the generated patterns are shown in fig 11

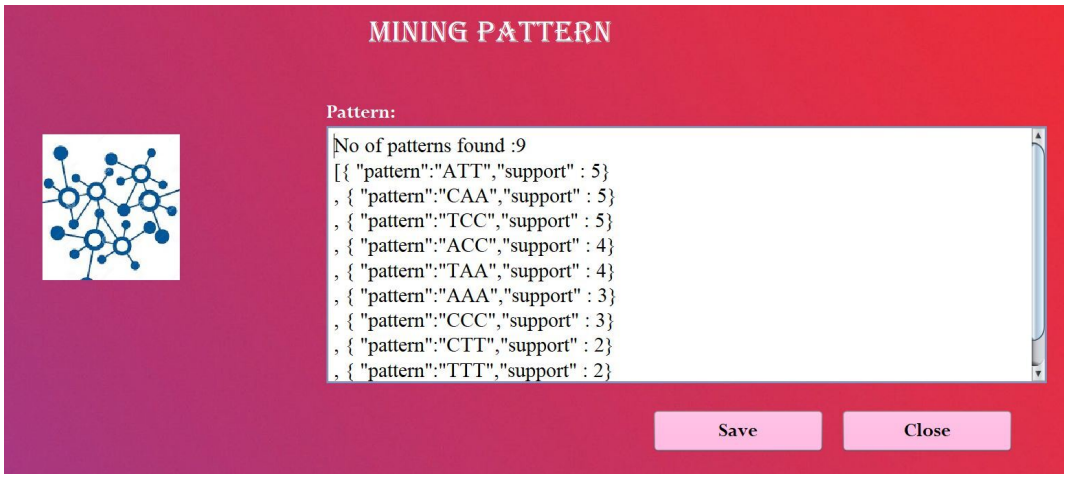

Figure 11: Pattern Discovery from Input Results

What if we select the second option shown in Fig.12 which would be Pattern Discovery from Data Set. So, in order to identify NMSP patterns, we must first choose the type of data set we want to analyze. Providing a minimum and maximum gap for pattern discovery is the next stage. Then we have decide whether to move forward using the minimal support need or the pattern count and size. We will receive all of the various sorts of patterns and their support values after providing all of the inputs. We will analyze all possible findings for both methods in the Evaluation section.

Just in same case of Pattern Discovery from Data Set we can see the demo results shown in fig 13

Figure 12: Pattern Discovery from Data Set

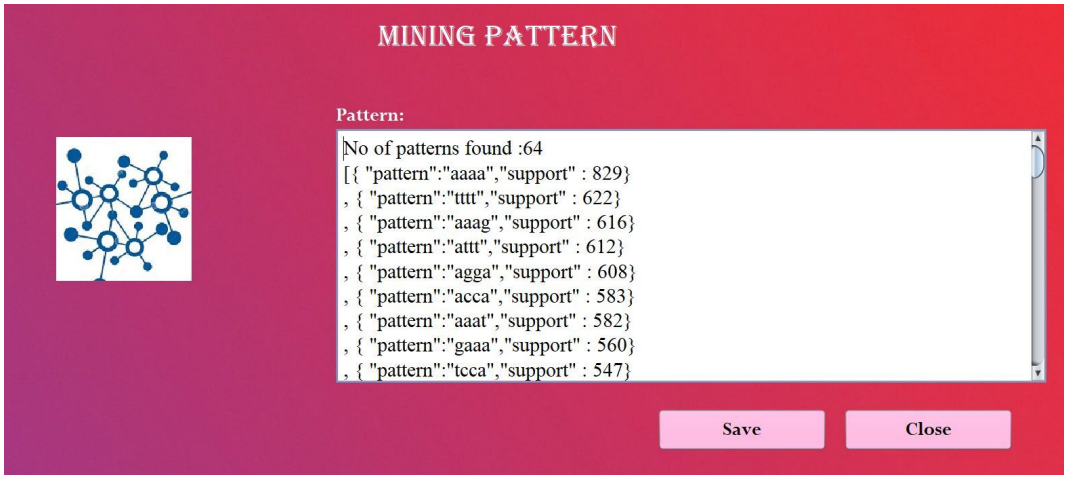

Figure 13: Pattern Discovery from Data Set Results

6 Evaluation

This section will evaluate the experiments performed by using existing and proposed algorithms. The experiments would be focused upon processing memory, top k count discovering candidate patterns, as well as processing time. Using a minimum threshold, the current approach mainly focuses on determining candidate patterns for NMSP. However, one main disadvantage of this approach is that when we reduce the minimum support value, the processing time simultaneously decreases. So, when we define a minimum threshold it will filter the patterns at the time of candidate generation process. However, one main disadvantage of this approach is that when we reduce the minimum support value, the processing time simultaneously decreases. So, when we define a minimum threshold it will filter the patterns at the time of candidate generation process. Since there will be fewer patterns, also there will be fewer initial candidate patterns for NMSP.

6.1 Experiment 1: Pattern Extraction

This experiment is about extracting the patterns by using top k count method. We have tested the system on multiple data set and here we have enlisted two different data sets with two minimum support value and we are expressing the result of two data set. When we are executing our system with existing system i.e., minimum support value and when we change minimum support value the number of pattern count is changing. As we increase the minimum support value simultaneously pattern count get decreases. When we define any pattern size we get list of pattern contain with different pattern sizes. Therefore, we cannot put any constraint on pattern size. So, by using the proposed method when user enters top k pattern count these many pattern retain to the user shown in fig 14. Along with minimum support value user can define pattern size so, for that size user can get top k matching patterns.

DB/Mini	Min Sup 100	Min Sup 500	top k 100	top k 200
Baby 1	Pattern found: 939	Pattern found: 194		
	Pattern Description:	Pattern Description:		
	size 4 = 37	size 18 = 1	size n= 100	size n = 200
	size 5 = 231	size 3 = 4	where n = {4-30}	Where n = {18, 3,4,5,6,12}
	size 6 = 341	size 4 = 49		
	size 7 = 9	size 5 = 83		
	size 8 = 6	size 6 = 47		
	size 9 = 5	size 12 = 10		
	size 10 = 5			
	size 12 = 222			
	size 14 = 3			
	size 15 = 2			
	size 18 = 66			
	size 20 = 1			
	size 21 = 2			
	size24 = 8			
	size 30 = 1			
DNA 1	Pattern found: 119	Pattern found: 22		
	Pattern Description:	Pattern Description:	size n = 100	size n = 200
	size 4 = 11	size 3 = 7	where n = {4,5,6}	where n = {3 & 4 }
	size 5 = 106	size 4 = 15		
	size 6 = 2			

Figure 14: Extracted Pattern with there counts and size

6.2 Experiment 2: Comparison of Processing Memory

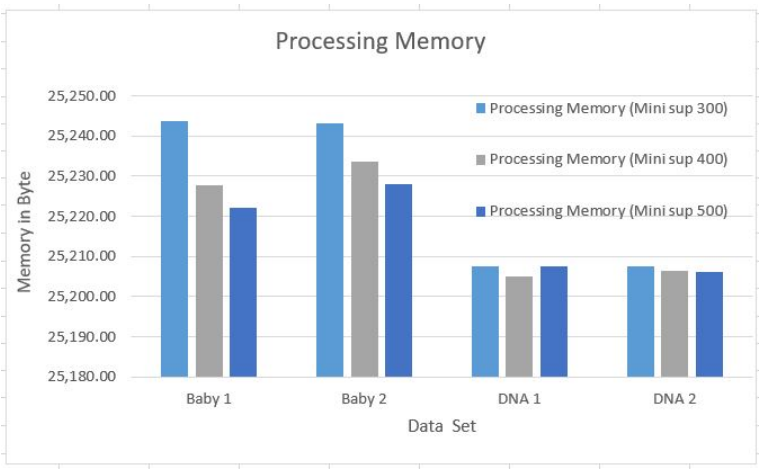

Figure 15: Processing Memory using Minimum Support (Existing System)

In this experiments we will be evaluating how much memory is used while processing different operation based on minimum support and top k count method. So, we can get a observation from fig 15 of how high an memory processing could go while identifying NMSP candidate patterns. Basically to examine this experiment the data set used is of Baby data set and DNA data set. So, if we increase the minimum support value the memory usage decreases and it same goes for top k pattern. So, while using the current technique we got 25.246 byte (i.e., 0.025246 megabyte) and the lowest we got is 25,200(i.e., 0.0252 megabyte). By using top k count shown in fig 16 the highest memory processing we got is 25,000 byte (i.e. 0.025 megabyte) and the lowest we got is 20,000 (i.e., 0.02 megabyte).

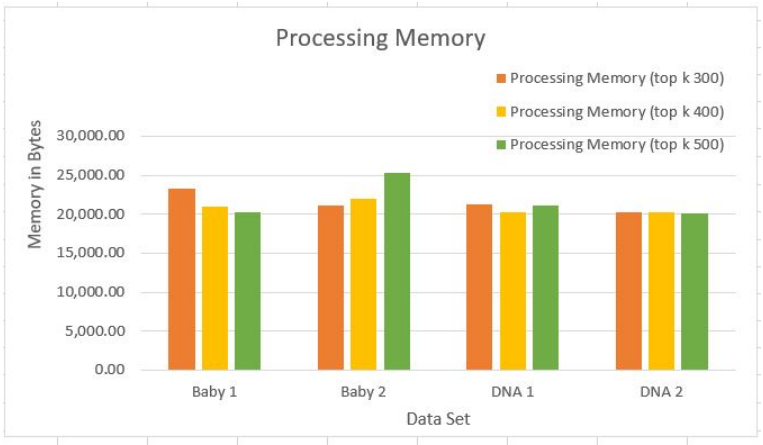

Figure 16: Processing Memory using Top k Count (Proposed System)

6.3 Experiment 3: Comparison of Processing Time

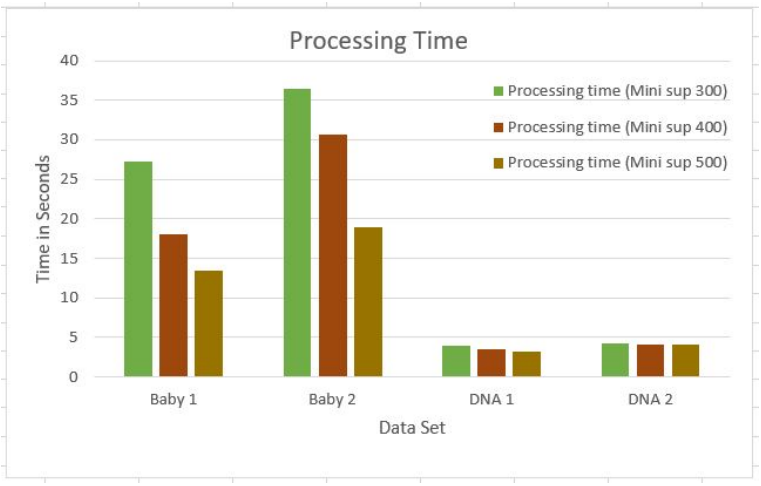

Figure 17: Processing time using Minimum Support (Existing System)

In experiments we are analyzing the processing time with existing and proposed method. So, above fig 17 shows that when we try to increase the minimum threshold value the processing time increases and when we try to reduce the minimum threshold value the processing time decreases but chances of getting multiple pattern will be filter before we get candidate patterns. In both of fig 17 and fig 18 we are using Baby data set and DNA data set. With the help of these data set we are finding non-overlapping pattern and hence, we are able to analyze how much of processing time is required by using both existing and proposed method. By using minimum threshold the highest processing time we required was between 35 to 40 seconds and by using top k pattern we got 27 to 30 seconds.

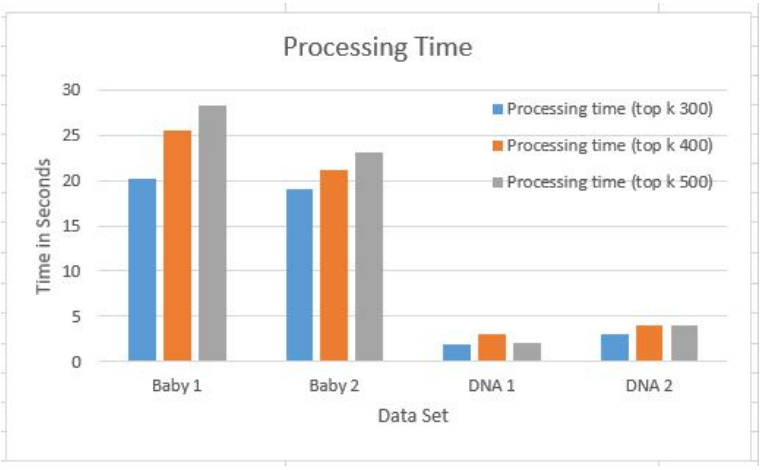

Figure 18: Processing time using Top k Count(Proposed System)

6.4 Discussion

This project implements the NetNMSP algorithm where the existing algorithm works on minimum support value. So, the minimum support works on finding the candidate patterns for NMSP i.e., Non-overlapping Maximal Sequential Pattern. But the existing system has an dependency issue which is minimum support value. So, based on existing method i.e., minimum support when the system provide a higher value as minimum threshold then only we get large candidate patterns but the processing time also increases. If we reduce the threshold then we get low candidate patterns and processing time also decreases. The filtration of pattern was not efficient. But by using top k count (i.e., proposed system) we can give high pattern count and we can also manage the number of pattern to be found by using pattern size and this technique also reduces processing time. The top k pattern require less time and the reason is because we are defining the pattern size. So, any chances of getting multiple pattern will be filter before we get candidate patterns. To validate the proposed system we have conducted 3 experiments for comparing existing and proposed system results.The experiments are based on pattern extraction, processing time and processing memory. So, to consider experiment 1 when user gives the input as a top k pattern count and pattern size they will get different types of candidate pattern and with that they will get what is the size of pattern and total number of count. Looking at experiment 2 and 3 it describer the comparison of processing memory and time between existing and proposed system. We can see that proposed system findings are showing good results for both of the experiments and its shows how efficient top k count method works depending upon processing time and processing memoroy.

7 Conclusion and Future Work

The selection criteria were based on the research problem we would be addressing. Therefore, the aim of this project's research was to identify the best or most effective approach for employing the top k count method to extract non-overlapping patterns from a series of sequential patterns. Sequence pattern mining was frequently used in a variety of fields and seems to have the appearance of a huge number of patterns. Therefore, implementing top-k pattern mining, which essentially includes identifying the significantly higher frequent k pattern, would turn out to be a successful technique. In this one of the most difficult factors were patterns occur more frequently but by using top k count method we been able to resolve that and as well as it also reduces the dependency of minimal support value. The term "non-overlapping maximal sequence pattern (NMSP) mining" refers to the discovery of common patterns with rare sub-patterns. The problem of using the existing method to detect NMSP patterns with minimal help was that it might even be challenging for a data analyst of this caliber to set a suitable specified threshold. Getting it hard to understand the actual patterns could result from using the incorrect minimum support threshold. The top k count approach used in this project has so far changed the current methodology. Using top-k patterns and identify patterns may also help to reduce the probability of getting inefficient processing time. It will therefore be effective to discover the unique patterns with order of preference by eliminating the minimal support dependence. The overall proposed system is fully functional and for now the data set which was used for implementation was static and it didn't have any null value so, no prepossessing was required but if we try to implement this project with

dynamic data i.e., real time data its efficiency might get reduce. But to consider this as an challenge we can take this limitation as future work and also the drawback of the real time data set can be is resolved. We can also improve its working by using cloud server. The cloud server can replace all the minimum requirements and different type large scale data set can be implemented very efficiently.

References

Bharati, M. and Ramageri, M. (2010). Data mining techniques and applications.

Chai, X., Yang, D., Liu, J., Li, Y. and Wu, Y. (2018). Top-k sequence pattern mining with non-overlapping condition, *Filomat* **32**(5): 1703–1710.

Chee, C.-H., Jaafar, J., Aziz, I. A., Hasan, M. H. and Yeoh, W. (2019). Algorithms for frequent itemset mining: a literature review, *Artificial Intelligence Review* **52**(4): 2603–2621.

Dahbi, A., Balouki, Y. and Gadi, T. (2017). Using multiple minimum support to auto-adjust the threshold of support in apriori algorithm, *International Conference on Soft Computing and Pattern Recognition*, Springer, pp. 111–119.

Fournier-Viger, P., Lin, J. C.-W., Kiran, R. U., Koh, Y. S. and Thomas, R. (2017). A survey of sequential pattern mining, *Data Science and Pattern Recognition* **1**(1): 54–77.

He, R., Dobie, F., Ballantine, M., Leeson, A., Li, Y., Bastien, N., Cutts, T., Andonov, A., Cao, J., Booth, T. F. et al. (2004). Analysis of multimerization of the sars coronavirus nucleocapsid protein, *Biochemical and biophysical research communications* **316**(2): 476–483.

Hikmawati, E., Maulidevi, N. U. and Surendro, K. (2021). Minimum threshold determination method based on dataset characteristics in association rule mining, *Journal of Big Data* **8**(1): 1–17.

Jiang, H. and Meng, H. (2017). A parallel fp-growth algorithm based on gpu, *2017 IEEE 14th International Conference on e-Business Engineering (ICEBE)*, IEEE, pp. 97–102.

Jiawei, H., Micheline, K. and Jian, P. (2011). Data mining concepts and techniques third edition, *The Morgan Kaufmann Series in Data Management Systems* **5**(4): 83–124.

Karthik, P. and Saira Banu, J. (2020). Frequent item set mining of large datasets using cuda computing, *Soft Computing for Problem Solving*, Springer, pp. 739–747.

Li, Y., Zhang, S., Guo, L., Liu, J., Wu, Y. and Wu, X. (2022). Netnmsp: Nonoverlapping maximal sequential pattern mining, *Applied Intelligence* pp. 1–24.

Pan, J.-S., Lin, J. C.-W., Yang, L., Fournier-Viger, P. and Hong, T.-P. (2017). Efficiently mining of skyline frequent-utility patterns, *Intelligent Data Analysis* **21**(6): 1407–1423.

Phan, H. (2019). An efficient mining algorithm of closed frequent itemsets on multi-core processor, *International Conference on Advanced Data Mining and Applications*, Springer, pp. 107–118.

Ryang, H. and Yun, U. (2015). Top-k high utility pattern mining with effective threshold raising strategies, *Knowledge-Based Systems* **76**: 109–126.

Salam, A. and Khayal, M. (2012). Mining top- k frequent patterns without minimum support threshold, *Knowledge and information systems* **30**(1): 57–86.

Trivedi, J. and Patel, B. (2017). An automated support threshold based on apriori algorithm for frequent itemsets, *Int J Adv Res Innovative Ideas Educ* **3**(6): 446–52.

Wang, F., Dong, J. and Yuan, B. (2013). Graph-based substructure pattern mining using cuda dynamic parallelism, *International conference on intelligent data engineering and automated learning*, Springer, pp. 342–349.

Wang, T., Duan, L., Dong, G. and Bao, Z. (2020). Efficient mining of outlying sequence patterns for analyzing outlierness of sequence data, *ACM Transactions on Knowledge Discovery from Data (TKDD)* **14**(5): 1–26.

Wang, Y., Wu, Y., Li, Y., Yao, F., Fournier-Viger, P. and Wu, X. (2022). Self-adaptive nonoverlapping sequential pattern mining, *Applied Intelligence* **52**(6): 6646–6661.

Wu, Y., Tong, Y., Zhu, X. and Wu, X. (2017). Nosep: Nonoverlapping sequence pattern mining with gap constraints, *IEEE transactions on cybernetics* **48**(10): 2809–2822.

Wu, Y., Wu, X., Min, F. and Li, Y. (2010). A nettree for pattern matching with flexible wildcard constraints, *2010 IEEE International Conference on Information Reuse & Integration*, IEEE, pp. 109–114.

Zhang, S., Wu, X., Zhang, C. and Lu, J. (2008). Computing the minimum-support for mining frequent patterns, *Knowledge and Information Systems* **15**(2): 233–257.

Zhou, J., Yu, K.-M. and Wu, B.-C. (2010). Parallel frequent patterns mining algorithm on gpu, *2010 IEEE International Conference on Systems, Man and Cybernetics*, IEEE, pp. 435–440.