Abstract
Software companies employ analysts to reveal and understand user behavior in log data. Analysts are interested in identifying operation groups that represent meaningful tasks performed by users inside applications. This is challenging, due to the amount of noise in log data, namely, operations can be repeated and unnecessary operations are often performed for a given task. In this project, I introduce a novel energy based frequent pattern ranking technique that extracts frequent user tasks from application logs by modeling the affinity between various sets of operations. Due to the lack of a real application log, a synthetically generated application log that mimics real application logs was used. My experimental study shows my model offers higher precision and recall in identifying frequent user tasks compared to state of art methods.

Introduction
Application log data contains valuable information about user behavior that can inform technical or business decisions. According to a major software company, identifying meaningful and frequent user tasks is an important milestone in many log analysis scenarios, such as recognizing what features needs to be prioritized in product roadmaps. While identifying frequent user task is important, doing so for large-scale, complex log data is difficult due to the volume of data, complexity of the data, and perhaps most importantly, diversity in user behavior and domain context. Currently, identifying frequent user tasks involves using frequent pattern mining and sequential pattern mining. [3, 4, 5, 8] These techniques alone are not ideal in addressing some of the challenges in task mining. My work is inspired by the idea of patterns cohesion, which prioritizes the patterns whose events appear contiguously in the supporting sequences with no or few outliers (events not belonging to the pattern). In my model, I attempt to expand upon this idea by combining it with association mining techniques to form an energy based score for ranking frequent patterns.

Related Work
A major problem with using frequent itemset mining and sequential mining techniques on log data to identify frequent user tasks is that such techniques often generate a huge number of patterns. [2] Even for strict classes such as closed patterns, the results can be overwhelming. To introduce my energy based approach, we shall first introduce a few concepts.

Fig. 1 below shows instances of task 1 (T1), a version management strategy that involves creating a new file from an existing one via copy and paste. Task 2 (T2) involves cropping an image and checking the size of its dimensions. The instances of T1 are shown in red, with red vertical lines outlining the instance boundaries. The instances of T2 are shown in blue, with blue vertical lines outlining the instances boundaries. [1]

The first two event sequence in Fig. 1 shows that T1 can be accomplished using different orderings of operations. The third and fourth event sequence in Fig. 1 show that T2 can be accomplished using different orders of operations as well. The fifth and sixth event sequence show repetition of operations for T1 and T2. The seventh and eighth event sequence show that T1 and T2 can be performed in a single session. It’s important to note that in all cases, users are executing the steps of a task.
contiguously, without outliers (unnecessary operations).

**FIG. 1**

We shall review concept from frequent pattern mining since it serves as the foundation for our work.

**DEFINITION 1.** An event sequence \( S = [E_1, E_2, \ldots, E^n] \) \((E^i \in E)\) is an ordered list of events, where \( E \) denotes the event dictionary and \( i \) denotes the order of event \( E^i \) in \( S \).

**DEFINITION 2.** A sequence database \( D = \{S_1, S_2, \ldots, S_n\} \) is an unordered set of sequences.

**DEFINITION 3.** In our discussion, a pattern \( P \) is either (i) a set of events whose members appear in random order, or (ii) a sequence of events that appear as subsequences(s), in one or more sequences in an event sequence database.

**DEFINITION 4.** The support set \( D_P \) of a pattern \( P \) in a sequence database \( D \) is the largest subset of \( D \) where \( P \) appears in all sequences belonging to \( D_P \). The support of \( P \) is quantified as the percentage ratio of the size of \( D_P \) and \( D \).

**DEFINITION 5.** Frequent patterns \( F = \{P_1, P_2, \ldots, P_j\} \) is a set of patterns (of same type) where the support of each pattern in a given database is no less than a user-specified threshold \( \Theta \).

**DEFINITION 6.** The occurrence window \( W_{PS} \) of a pattern \( P \) in a sequence \( S \) refers to the interval(s) within \( S \) that contains \( P \).

**DEFINITION 7.** The minimum length occurrence window or minimum occurrence window \( W_{PS}^{L-} \) of a pattern \( P \) in a sequence \( S \) refers to the minimum length interval(s) within \( S \) that contains \( P \). Here, the function \( L() \) returns length and the superscript \( (L-) \) denotes minimum length.

\[
W_{PS}^{L-} = \arg \min_{W_{PS}} L(W_{PS}) \tag{1}
\]

It is important to note that frequent patterns can be either sets or sequences of events. In frequent itemsets, ordering is not important while in sequential patterns, ordering is important. [3]

**Fig. 2(a)** shows five sequences and **Fig. 2(b)** shows different types of patterns in the sequences with \( >=40\% \) support.
Cohesive (no outliers) itemsets are set of items such that their minimum length occurrence window does not contain items not belonging to the itemset. For instance, in FIG. 2, \{A, B, C, D\} is a cohesive itemset since in S3 and S5, there are no items not belonging to \{A, B, C, D\}. Cohesion is a central concept to our energy based model.

Order-sensitive patterns such as sequential patterns and episode suffer drawback in log mining since a task can be completed in different orderings of operations. Cohesion-insensitive patterns such as frequent itemsets and sequential patterns cannot distinguish between the set of events or operations that appear adjacently and the set of operations that appear randomly in supporting sequences. For example, certain popular operations may appear in many sequences for different tasks. For those reasons, others have investigated the idea of membership base cohesion for frequent patterns.[1]

We shall introduce the item of outlier based minimum occurrence window as that is a central concept to previous work [1] and our energy based model:

For example, suppose the pattern is \{A, B, C\} and the sequence is \{A, B, B, B, C, W, A, C, F, B\}. Two possible intervals in the sequence that contains \{A, B, C\} are \{A, B, B, B, C\} and \{A, C, F, B\}. The minimum length occurrence window is \{A, C, F, B\} since this interval only has length of 4. However, the outlier based minimum occurrence window is \{A, B, B, B, C\} since it has zero outliers while \{A, C, F, B\} contains one outlier (F).

Previous work [1] has attempted to rank patterns based on cohesion score, which is the difference between the length of a pattern and the median count of outliers in their outlier based minimum occurrence windows across all sequences. This is based on the assumption that an interval corresponding to a task should have minimal amount of outliers compared to a random interval. However, ranking patterns solely based on cohesion does not take into account of ordering of operations in a task or other contextual information. While some tasks can be performed with different orderings of operations, for many tasks there are specific sequences of operations that must be followed. Furthermore, ignoring ordering or contextual information can lead to instances where a highly cohesive pattern that is not a task gets ranked highly. As an example, suppose \{A, B, C\} are operations for a task \(T_1\), and \{D, E, F\} are operations for another task \(T_2\). However, \{B, A, D\} is not a task. Consider the following sequences:

S1: \{C, B, A, D, E, F\}
S2: \{C, A, B, D, F, E\}
S3: \{C, A, B, X, Y, Z, G, T, U\}

\[ W_{PS}^{(O-)} = \arg \min_{W_{PS}} O(W_{PS}) \]
S4: [N, M, Y, T, D, E, F]

In S1, the user performs T1 and then performs T2. In S2, the user performs T1, and then T2, albeit in a different order of operations for both tasks compared to S1. In S3, the user performs T1, followed by some other random tasks, and in S4, the user performs others tasks followed by T2.

Suppose the minimum support is set to 2 for the initial pattern mining, \{B, A, D\} will be mined and because of its high cohesion in both S1 and S2, it will be ranked highly even though it is not a task.

**ENERGY BASED PATTERN RANKING**

Energy based pattern ranking is inspired by the interaction of atoms and molecules. Individual atoms can behave differently depending on whether they are free individual atoms or part of a molecule. As an example, while hydrogen atom (H+) is highly flammable, water (H2O) is not. We combine this thought with the techniques of association mining (Jiawei) to measure the energy of a pattern. A pattern with a high likelihood of being a task would have high energy.

Concept 1: The energy of an element in an interval is the sum of its left and right energies.

\[
\text{Energy}_{\text{element}} = \text{Energy}_{\text{left}} + \text{Energy}_{\text{right}}
\]

The left energy of an element in an interval is best illustrated through an example. Suppose we have the sequence [A, B, C, D, E, F, G], and we want to calculate the left energy of element D. To do so, we consider the confidence of association rule (C > D). In another word, (C > D) is the percentage of times that D follows C, given all occurrences of C. Similarly, (B,C > D) is the percentage of times that D follows B,C, given all occurrences of sequence B,C. Finally, define (A, B, C > D) as the percentage of times that D follows sequence A, B, C.

We then define the \(\text{Energy}_{\text{left}}\) of D as:

\[
\max \left( (C>D), (B,C>D), (A,B,C>D) \right)
\]  (equation 1)

The left energy attempts to capture the contextual relationship, or affinity, of D with the elements to the left of it. While D may have a weak affinity with C, it might have a strong affinity with B,C, and by taking the maximum of \( (C>D), (B,C>D), (A,B,C>D) \), we can capture the strongest possible affinity. For elements in the beginning of a sequence, which would be A in the above example, the left energy is undefined and will be ignored in future calculations.

For elements in the second position from the left, which would be B in the above example, the left energy is simply \( (A>B) \). For elements in the third position from the left, which would be C in the above example, the left energy is defined as \( \max( (B>C), (A,B>C) ) \). For all other elements, their left energy is as defined in equation 1.

In most cases, the confidence of association rule between a single element to another single element will be weak, and in a large log with relatively few distinct operations, it will converge around a fixed value approximating

\[
\frac{1}{\text{number of unique operations}}
\]

By exploring more complex relationships, and taking the maximum affinity among the relationships, we can better differentiate between operations essential to a task and outliers.

For instance, consider the previous example, where T1 is highlighted in blue and T2 is highlighted in pink:

S1: [C, B, A, D, E, F]
S2: [C, A, B, D, F, E]
S3: [C, A, B, X, Y, Z, G, T, U]
S4: [N, M, Y, T, D, E, F]

Again, we consider the pattern \{B, A, D\}, which is not a task. Its outlier based minimum occurrence window in S1 is [B, A, D]. We should expect that the left energy of D would be weak compared to the left energy of B in [C, A, B], since \{A, B, C\} is a task and its elements should appear together more frequently.
in any order compared to that of \{B, A, D\}. Therefore \{B, A, D\} will be ranked low based on the energy based metric.

Analogously, we can also define \textit{Energy\_right} for an element. As an example, the right energy of D in sequence \[A, B, C, D, E, F, G\] is defined as

\[
\max [ (D<E), (D<E,F), (D<E,F,G) ] \quad \text{(equation 2)}
\]

\(D<E\) is the percentage of times that D preludes E, given all occurrences of E. Similarly, \((D<E,F)\) is the percentage of times that D preludes E,F, given all occurrences of sequence E,F. Finally, define \((D<E,F,G)\) as the percentage of times that D preludes sequence E, F, G. The right energy of elements near the right edge of a sequence are handled in a similar manner as that of left energy for elements near the left edge.

When calculating energies, we treat contagious elements of the same type as one single unit. For example, \((D<E,E,E) = (D<E,E) = (D<E)\), and \((A,C,C>B) = (A,C>B)\), and so on. This conforms with our definition of outlier based minimum occurrence window, where a sequence such as \[A,A,B,B,B,C\] and \[A,B,C\] all correspond to pattern \{A,B,C\} and has zero outliers. Therefore, they should have the same energy. This brings us to the next concept:

\textit{Concept 2: The energy of a sequence is the sum of the left and right energies of its elements divided by a normalization factor.}

\[
\text{Energy}_{\text{seq}} = \frac{\sum \text{Energy\_right} + \sum \text{Energy\_left}}{\text{normalization factor}}
\]

We normalize the energy of a sequence by summing the left and right energies of its elements and dividing by the total number of energies summed. It’s important to remember that the two edge elements only have one (either left or right) type of energy while every other elements has two types of energy. We also treat contagious elements of the same type as one element.

By defining energy this way, the model can accommodate repeated operations in a task, while also capturing the intricate relations among tasks. For tasks where order of operation is important, the left and right energies will be affected accordingly. Even among tasks where order of operation is unimportant, operations that often cluster together will tend to have higher energies compared to that of a randomly distributed sequence of operations.

To calculate the energy of a pattern, we need to consider the energies of all outlier based minimum occurrence window in which the pattern occur across all sequences in a log. This brings us to concept 4.

\textit{Concept 4: A pattern’s energy is the defined as the mean energies of all its outlier based occurrence window across all sequences.}

\[
\text{Energy}_{\text{pattern}} = \frac{\sum_{\text{seq}\text{= all olbmov}} \text{Energy}_{\text{seq}}}{\text{number of olbmov}}
\]

*olbmov stands for outlier based minimum occurrence window.

In reality, experiments show that the model works better if we limit ourselves to consider only cases where the outlier based occurrence window contains no more than a certain number of outliers. For instance, consider the following sequence:

\[A, B, D, E, F, F, G, H, C\].

Suppose \{A,B,D\} is a task and so is \{E,F,G,C\}. Their respective outlier based minimum occurrence window are \[A, B, D\] and \[E, F, G, H, C\]. In this case, the occurrence of outlier \(H\) in \[E, F, F, G, H, C\] will negatively affect the energy of the outlier based minimum occurrence window \[E, F, F, G, H, C\], which in turn negatively affects the energy of the pattern \{E,F,G,C\}. This makes sense because the more outliers there are in a sequence/window, the less likely that sequence is a task. In this case, the outlier acts as noise which decreases the chance of pattern \{E,F,G,C\} being classified as a task, when it in fact is. Outliers are unavoidable due to various factors, such as when an user accidentally presses the wrong key when performing a task, but the...
**energy based method can minimize the impact of a small amount of outliers.** For example, suppose \{A, B, C, D, F\} is a task and it appears in sequence [K, F, D, E, C, A, B, G]. The corresponding outlier based minimum occurrence window is [F, D, E, C, A, B]. Element C will have a weak affinity with outlier E, but it will have stronger than normal affinity with F and D, so (F,D,E>C) will be higher than expected despite the presence of an outlier.

On the other hand, suppose \{B,D,C\} is a task, and in sequence [A, B, D, E, F, G, H, C], it is not performed. The colored elements in this sequence belong to tasks \{A, B, D\} and \{E, F, G, C\}. However, all the elements of \{B, D, C\} appear in the sequence and the corresponding outlier based minimum occurrence window is [B, D, E, F, G, H, C]. When calculating the overall energy of pattern \{B, D, C\}, the low energy of [B, D, E, F, G, H, C] will bring down the ranking of pattern \{B, D, C\} despite having nothing to do with the task. Therefore, for each pattern, I only consider its outlier based minimum occurrence window if the number of outliers is below a certain amount. Since the probability of outlier occurrence is proportional to window length, I use a parameter $\alpha$ to specify the outlier threshold, as a percentage of the length of the outlier based minimum occurrence window.

In my experiment, I found that $\alpha$ of 0.2 works the best.

To summarize, I made a few assumptions when formulating my energy based pattern ranking approach:

1. Most tasks are performed with little to no outliers and only outlier based minimum occurrence windows with outliers below a certain threshold should be considered in order to best capture said tasks. This is more effective than utilizing metrics that uses the median number of outliers across all sequences in a given patterns, like in some previous works.
2. In case that outliers do occur, provided they are small in number, the energy based model can minimize the adverse effect of the outliers.
3. Single element to single element association rules like (C>A) do not provide much information if the number of unique operations is few relative to the length of the log. In patterns corresponding to tasks, multi-elements to single element association rules like (B, C>A) will certainly dominate in the energy calculations. However, the inclusion of single element to single element association rules are still valuable in cases where the multi-element to single element association rules have extremely low confidence, such as near the boundaries between tasks. For example, suppose a sequence [A, B, C, D, E, F] contains task \{A, B, C\} and \{D, E, F\}. Furthermore, suppose the association rule (A,B,C>D) has confidence of zero, which gets ignored, the left energy of D will then be (C>D), which prevents the energy of (D, E, F) from being severely affected by the boundary condition. The argument would also apply if A, B, C are outliers.
4. Tasks can often contain contiguous repeated operations and the energy model takes that
into account by treating contiguous elements of the same type as a single element.

5. In certain tasks, ordering is important while in other tasks, order is more random. The energy model excels in discriminating tasks where ordering is important but even in tasks where ordering can be random, closely clustered operations will still lead to a higher energy than normally expected. As an example, suppose \( \{A, B, C, D\} \) is a frequent pattern, we would expect \((B, A, C > D)\) or \((C, D, A > B)\) to have higher confidence level than a random association rule such as \((X, Y, Z > T)\).

**EXPERIMENTAL RESULTS**

The synthetic log data is generated using 18 different tasks and each task contains 3 or 4 different operations. Some tasks have ordering constraints among all or part of operations while others have no ordering constraints. There were 17 different types of operations. Furthermore, the length of each task is limited to 10 operations (allowing for repeated operations) and each sequence is limited to at most 10 tasks. There are 1000 sequences in the log.

We compare our result with Himel’s cohesion score [1] approach for ranking frequent tasks, which is the most recent work in log mining that I am aware of, and is published in 2017 ACM International Conference on Intelligent User Interface (IUI).

The evaluation metric we use is precision and recall for the top 20 ranked patterns. Unfortunately, a direct comparison is difficult because Himel’s work is done using a real user logs from a major software company, with no ground truth labels of tasks. Rather, a user study is conducted where top ranked patterns are rated by 10 participants from a score of 0 to 5 on whether the pattern is likely a task. For each pattern, the mean score is calculated and those with mean score above 3.0 are deemed tasks.

To make a valid comparison, I implemented Himel’s cohesion score ranking approach on the same synthetic log used for my energy based approach. For both cases, I mined frequent patterns using a threshold of 10%, which resulted in roughly 5500 patterns. The precision and recall for the top k patterns are shown below, where k is maximum of 20.

<table>
<thead>
<tr>
<th>My Results</th>
<th>Himel’s Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>Recall</td>
</tr>
<tr>
<td>1.0</td>
<td>0.055</td>
</tr>
<tr>
<td>1.0</td>
<td>0.111</td>
</tr>
<tr>
<td>0.66</td>
<td>0.111</td>
</tr>
<tr>
<td>0.5</td>
<td>0.111</td>
</tr>
<tr>
<td>0.6</td>
<td>0.166</td>
</tr>
<tr>
<td>0.5</td>
<td>0.166</td>
</tr>
<tr>
<td>0.438</td>
<td>0.166</td>
</tr>
<tr>
<td>0.375</td>
<td>0.166</td>
</tr>
<tr>
<td>0.33</td>
<td>0.166</td>
</tr>
<tr>
<td>0.3</td>
<td>0.166</td>
</tr>
<tr>
<td>0.27</td>
<td>0.166</td>
</tr>
<tr>
<td>0.25</td>
<td>0.166</td>
</tr>
<tr>
<td>0.231</td>
<td>0.166</td>
</tr>
<tr>
<td>0.214</td>
<td>0.166</td>
</tr>
<tr>
<td>0.2</td>
<td>0.166</td>
</tr>
<tr>
<td>0.1875</td>
<td>0.166</td>
</tr>
<tr>
<td>0.176</td>
<td>0.166</td>
</tr>
<tr>
<td>0.166</td>
<td>0.166</td>
</tr>
<tr>
<td>0.158</td>
<td>0.166</td>
</tr>
<tr>
<td>0.15</td>
<td>0.166</td>
</tr>
</tbody>
</table>
From the results, my energy based approach was able to capture 3 of the 18 frequent user tasks in the top 20 results. Amazingly, the two top ranked patterns are two of the 18 tasks.

Himel’s cohesion based method was not able to capture any of the 18 frequent user tasks in the top 20 results. The top ranked task from Himel’s model was ranked at 23rd. Upon investigation, the highly ranked patterns using the cohesion score approach are all relatively long patterns. This is not surprising since cohesion score is calculated by subtracting the median number of outliers for a pattern across all its outlier based minimum occurrence window from the length of the pattern.

Since analysts will manually examine the top ranked results from log mining, it is ideal to rank frequent tasks highly for easier analysis. In this regard, my energy based pattern ranking method outperformed the state of the art model.

**DISCUSSION**

The synthetic log data is generated using a carefully designed algorithm that mimics real user logs. My follow up work include investigating the frequent patterns that was not ranked in the top 20 results, and use any promising observations to further fine tune my model. Currently, I notice that among the highly ranked patterns that are not individual tasks, most are composed of two or more tasks with high “inter-task” affinity.

Due to time constraints, I was unable to explore whether including more elaborate association rules will further improve the performance. I plan to investigate as next steps the inclusion of longer association rules such as (A,B,C,D>E) and test whether longer, more elaborate rules may help in detecting rare and/or complex tasks. Once my model has performed sufficiently well on the synthetic data set, I will test it on a real user log.

It is worthy to mention that the most time consuming portion of the algorithm is calculating the various association rule confidences and placing them in a dictionary. The energy calculation for all frequent patterns is relatively fast once the dictionary is created. To speed up the calculation of association rules, Apriori algorithm can be used rather than brute force method. Since the Energy Based Pattern Ranking algorithm does not take into account association rules that fall below a minimum support threshold, using Apriori algorithm enable us to ignore longer association rules such as (A, B, C>D) if any of its shorter subset such as (B, C >D) does not satisfy the minimum support threshold. Therefore, the algorithm can be made efficient and scalable.

**IMPLICATIONS**

In addition to software design, the techniques discussed in this paper can also be used to model user behavior. For instance, [11] study web visitation behavior of customers by mining web logs.

Furthermore, ideas in log mining can be valuable in mining temporal event sequence data in arenas like digital marketing or E-commerce[7,10], user-workflow [6], and online education [9].

**CONCLUSION**

In this work, I proposed an algorithm, Energy Based Pattern Ranking, to rank patterns corresponding to frequent tasks by maximizing the energy, which is a measure of cohesion and affinity among elements of patterns. I borrowed tried and true data mining techniques such as frequent pattern mining, association rule mining, Apriori algorithm and combined them in a novel way to achieve state of the art performance on mining user logs. The algorithm is scalable and simple to understand, while making sense intuitively. The proposed approach can therefore be used as a reliable tool to accurately mining frequent tasks from user logs.
References


[2]. Charu C. Aggarwal and Jiawei Han. 2014. Frequent Pattern Mining. Springer Publishing Company, Incorporated.

[3]. Jiawei Han and Micheline Kamber. Data Mining: Concepts and Techniques Morgan Kaufmann Publishers, August 2000.


