Improving Learning Implicit User Interest Hierarchy with Variable Length Phrases

TR CS-2003-17

Hyoung-Rae Kim and Philip K. Chan
Department of Computer Sciences
Florida Institute of Technology Melbourne, FL 32901, USA
hokim@fit.edu, pkc@cs.fit.edu

ABSTRACT

A continuum of general to specific interests of a user called a user interest hierarchy (UIH) represents a user's interests at different abstraction levels. A UIH can be learned from a set of web pages visited by a user. In this paper, we focus on improving learning the UIH by adding phrases. We propose the VPF algorithm that can find variable length phrases without any user-defined parameter. To identify meaningful phrases, we examine various correlation functions with respect to well-known properties and other properties that we propose.

1. INTRODUCTION

Motivation: User Interest Hierarchy (UIH) is a hierarchy of topics of interests from web pages visited by a user to provide a context for personalization [11]. More general interests, in some sense, correspond to longer-term interests, while more specific interests correspond to shorter-term interests. These help to identify the appropriate context underlying a user's behavior, which is important in more accurately pinpointing her interests. This can help rank the results returned by a search engine. Furthermore, the UIH provides a context to disambiguate words that could have multiple meanings in different contexts. For example, "java" is likely to mean the programming language, not the coffee, for a UIH that is learned from a user who has been reading computer science related pages. So this helps a user in searching relevant pages on the web. There are several ways to improve the UIH: using phrases, devising better clustering algorithm, finding more desirable threshold etc. In this paper we focus on using phrases.

Using phrases, in addition to words, we can improve the UIH. A term composed of two or more single words (called "phrase") usually has more specific meaning and can disambiguate related words. For instance, "apple" has different meanings in "apple tree" and in "apple computer".

Correlation functions and correlation thresholds are important components of a phrase finding algorithm. Tan et al. [16] suggested that there is no correlation function that is better than others in all application domains. This is because different correlation functions have different intrinsic properties, some of which may be desirable for certain applications but not for others. In order to find the right correlation function, one must match the desired properties of an application against the properties of the existing correlation functions. There are several key properties one should examine in order to select the right measure for a given application domain [12,16]. That is, we need to understand the desirable properties of correlation functions in general and in phrase finding.

Problem: We desire to find phrases to enrich the vocabulary for building more meaningful and accurate UIH's. The phrase finding algorithm should not restrict the maximum length of a phrase. Additional issues involve how to select a correlation function between words for the algorithm and how to determine an appropriate threshold for the correlation values.

Approach: In this paper we propose a variable-length phrase finding algorithm called VPF, which does not restrict the maximum length of a phrase. To identify meaningful phrases, we examine the desirable properties of a correlation function and propose additional desirable properties. The evaluation of our techniques is based on real data collected from our departmental web server.

The specific contributions of this work are:

1) We improve the UIH by adding phrases to the vocabulary.

2) We propose a variable length phrase finding algorithm (VPF) that does not restrict the maximum length of a phrase.

3) We identify new desirable properties for correlation functions in general and in phrase finding.

4) Our empirical studies indicate that VPF with the AEMI correlation function can match the most phrases identified by humans and VPF can improve the quality of UIH's.

The rest of this paper is as follows: Section 2 discusses related work in building a UIH [11], a phrase finding algorithm, and correlation functions; Section 3 introduces user interest hierarchies (UIH's) and phrases; Section 4 details our approach towards collecting phrases; Section 5 discusses our empirical evaluation; Section 6 summarizes our findings and suggests possible future work.

2. RELATED RESEARCH

A newsagent called News Dude [1], learns which stories in the news a user is interested in. The newsagent uses a multi-strategy machine learning approach to create separate models of a user's short-term and long-term interests. Unlike News Dude, DHC [11] group words (topics) into a hierarchy (a continuum of long-term to short-term interests) where more general interests are represented by a larger set of words. Each web page can then be assigned to nodes in the hierarchy for further processing in learning and predicting interests. It is like STC [18] which does not rely on a fixed vector of word features in clustering documents. Instead of, however, using a fixed user-provided threshold to differentiate strong from weak correlation values between a pair of words, DHC dynamically determined a reasonable threshold value. We add variable length phrases to the original words to improve a UIH.

Many concepts require more than single words to describe and characterize them. However, much research focuses on single words (unigrams) as features partly due to the large combination of possible multi-word phrases. Another reason is that earlier results from "syntactic" and "statistical" phrases were mixed [4]. We suspect that the ad hoc way of constructing statistical phrases [5] might have been a problem. Much of the statistical work in building multi-word features focuses on co-occurrence [3,13]. Zamir and Etzioni [18] introduced a method using a suffix-tree with linear time complexity, but this method accepts a threshold and can build phrases only based on the frequency – it cannot adopt other correlation functions. Chan's phrase finding algorithm [2] uses much more information but the main drawback is that the user has to specify the maximum phrase length. Our method finds variable length phrases and does not take a threshold. It can also use various correlation functions.

Tan et al. [16] demonstrated that not all measures are equally good at capturing the dependencies among variables and there is no measure that is consistently better than the others in all applications. They compared 21 existing correlation functions based on the key properties that Piatetsky-Shapiro [12] presented and other properties of a correlation function. We propose some additional desirable properties for a correlation functions.

3. USER INTEREST HIERARCHY AND PHRASES

A user interest hierarchy (UIH) organizes a user's general to specific interests. Towards the root of a UIH, more general (longer-term) interests are represented by larger clusters of words while towards the leaves, more specific (shorter-term) interests are represented by smaller clusters of words. To generate a UIH for a user, DHC accepts a set of web pages visited by the user as input. The web pages are stemmed and filtered by ignoring the most common words listed in a stop list [6].

Table 1 has a sample data set. Numbers in the left represent individual web pages; content has words stemmed and filtered through stop list. These words in the web pages can be represented by a UIH as shown in Figure 1. Each cluster node can represent a conceptual relationship, for example 'perceptron' and 'ann' (in italic) can be categorized as belonging to neural network algorithms, whereas 'id3' and 'c4.5' (in bold) in another node cannot. Words in these two nodes are mutually related to some other words such as 'machine' and 'learning'. This set of mutual words, 'machine' and 'learning', performs the role of connecting italic and bold words in sibling clusters and forms the parent cluster. We illustrate this notion in the dashed box in Figure 1.

One can easily identify phrases like "machine learning" and "searching algorithm" in the UIH, however only the individual words are represented in the UIH. By locating phrases from the pages, we can enrich the vocabulary for building the UIH. For example, the phrase "machine learning" can be identified and added to Pages 1-6.

Table 1. Sample data set

Page	Content
1	ai machine learning ann perceptron
2	ai machine learning ann perceptron
3	ai machine learning decision tree id3 c4.5
4	ai machine learning decision tree id3 c4.5
5	ai machine learning decision tree hypothesis space
6	ai machine learning decision tree hypothesis space
7	ai searching algorithm bfs
8	ai searching algorithm dfs
9	ai searching algorithm constraint reasoning forward checking
10	ai searching algorithm constraint reasoning forward checking

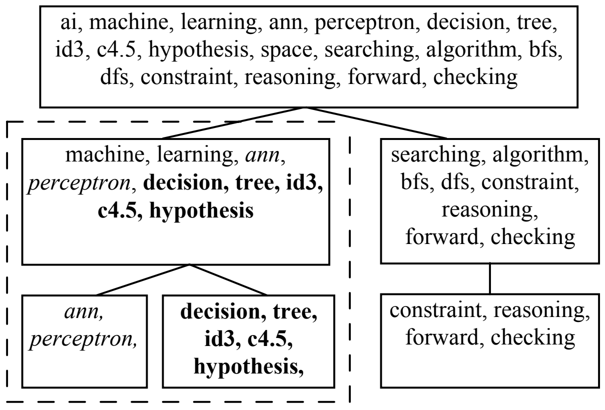
ai, machine, learning, ann, perceptron, decision, tree, id3, c4.5, hypothesis, space, searching, algorithm, bfs, dfs, constraint, reasoning, forward, checking
machine, learning, *ann*, *perceptron*, **decision, tree, id3, c4.5, hypothesis**
searching, algorithm, bfs, dfs, constraint, reasoning, forward, checking
ann, perceptron,
decision, tree, id3, c4.5, hypothesis,
constraint, reasoning, forward, checking

Figure 1. Sample user interest hierarchy

4. APPROACH

Our goal is to improve the UIH by using variable length phrases. In this section we focus on devising a phrase finding algorithm. Our proposed VPF algorithm consists of three components: main algorithm, correlation function, and threshold-finding method. We also examine the proper properties of correlation functions.

4.1 Main Algorithm

Our algorithm recursively combines words into phrases until it meets a stopping condition. Figure 2 illustrates the pseudo code for the VPF algorithm. In preparation for our phrase finding algorithm, we extract words from a web page visited by the user, filter them through a stop list, and stem them [6]. All distinct words are stored in PhraseList. Function FIND recursively collects longer phrases. PhraseLen keeps tracks of the length of the current phrase, which begins at 2. Using a correlation function, we calculate the strength of the relationship between a pair of words. We then build a weighted directed graph with each vertex representing a word and each weight denotes the correlation between two words. Since related words are more likely to be adjacent to each other in a document than unrelated terms, we measure co-occurrence of words in a document. Given the graph, called correlation matrix (CMatrix), VPF collects all pairs of words whose correlation value is larger than the threshold. The threshold is calculated only once by CalculateThreshold function, when the phrase length is 2. VPF recursively builds a correlation matrix with each element in a vertical row representing a phrase and with each element in a horizontal column representing a word, adding the pair of a phrase and a word into a PhraseList which correlation value is bigger than the threshold. The phrases between NOP (# of old phrases) and NCP (# of current phrases) in the PhraseList are collected from the previous CMatrix. They are used to build next correlation matrix in a CalculateCorrelationMatrix – new CMatrix between the new phrases and the original words are added into the previous CMatrix. Those pairs (one in a vertical line and the other in a horizontal line) which correlation value is bigger than the threshold compose new phrases, which are used to build new CMatrix. This recursive process stops when there is no increase in the # of NNP (# of next phrases).

```
VPF (Example, CORRELATION, FINDTHRESHOLD)
Example: A web page visited by the user.
CORRELATION: A function that calculates the "closeness" of
  two words.
FINDTHRESHOLD: A function that calculates the cutoff value
  for determining strong and weak correlation values.

1. Words are extracted from Examples, stemmed, and filtered
  through a stop list.
2. PhraseList ← distinct words with information of existing
  positions
3. SWordNum ← # of PhraseList (# of single words)
4. FIND (PhraseList, 0, 2, 0, SWordNum, SWordNum)
5. return PhraseList
```

```
FIND (PhraseList, Threshold, PhraseLen, NOP, NCP, NNP)
PhraseList: store all words and phrases
Threshold: differentiate "strong" from "weak" (initial value is 0)
PhraseLen: the length of the phrase that currently being generated
NOP: store # of old phrases (initial value is 0)
NCP: store # of current phrases (initial value is the # of single
  words)
NNP: store # of next phrases(initial value is the # of single words)

1. CMatrix ← CalculateCorrelationMatrix (NOP, NCP,
     INTERESTINGNESS, PhraseList)
2. if PhraseLen = 2 then
     Threshold ← CalculateThreshold (CORRELATION,
                   CMatrix)
   end if
3. for i = NOP to NCP-1 do
     for j=0 to SWordNum-1 do
       if CMatrix[i][j]>Threshold then
         make a phrase with i^th and j^th words in the PhraseList.
         revise the position information of the phrase.
         Add Phrase into the PhraseList.
         Increase NNP by 1.
       end if
     end for
   end for
4. if NCP = NNP then
     return Null
5. FIND (PhraseList, Threshold, PhraseLen+1, NCP, NNP, NNP)
```

Figure 2. Variable length phrase finding algorithm

As a phrase is collected, its existing positions are recounted. That is, as the length of a phrase increases, the probability of occurrence decreases. Using the same threshold value to all various length phrases has more consistent meaning than recalculating the threshold value again; this also guides the phrase finding algorithm to stop eventually.

Instead of using a fixed user-provided threshold to differentiate "strong" from "weak" correlation values, we try to dynamically determine a reasonable threshold (such as MaxChild [11] etc.) Since VPF collects only "strongly" correlated pairs, the lowest threshold value (like 0) yields the most number of phrases and the number of phrases reduces as the threshold increases. So, we calculate the threshold by simply averaging all the positive correlation values in a correlation matrix.

4.1.1 Related phrase finding methods

Related phrase finding algorithms include ones by Chan [2] and Zamir and Etzioni [18]. We compared their methods with VPF in six categories: time and space complexity, whether a user needs to specify the threshold or the maximum phrase length, flexibility (whether we can apply various correlation functions), and the amount of information (the number of correlations among words the algorithm calculates). Zamir's method has linear time complexity, which is faster than the others. Suppose the number of words is n, phrase length is m. The time complexity of Chan's method is $O(m^2 \times n^2)$, space complexity is $O(m^2 \times n^2)$. The time complexity of VPF is $O(n^2 + p_1 \times n + p_2 \times n + \ldots + p_{m-2} \times n)$ – that is equal to $O(n^2)$, where p is the number of collected phrases previously. The space complexity is $O(n^2)$. Both Zamir's method and Chan's method used user defined threshold value, but VPF used Average method. The critical disadvantage of Chan's method is that it has to get a user defined maximum phrase length. Since longer phrases can have more specific meaning we do not want to set the limitation on the length. The critical drawback of Zamir's method is that their algorithm can use only frequency information. They built the suffix-tree based on the overlap of words (frequency) and then collected neither too frequent nor too rare phrases. We also measured the amount of correlation information each of them use. Suppose we are collecting the phrase "a b c d". Zamir's method uses only the frequency of "a b c d". Chan's phrase collecting algorithm calculates the mutual values of "a b", "a c", "a d", "b c", "b d", and "c d". VPF calculates the mutual values of "a b", "ab c", and "abc d". Since VPF calculates the probability of "a", "ab", and "abc" respectively, it also uses almost the same amount of information.

Table 2. Comparison of phrase finding algorithms

Criteria	Zamir's	VPF	Chan's
Time	O(n)	$O(n^2)$	$O(m^2 \times n^2)$
Space	O(n)	$O(n^2)$	$O(m^2 \times n^2)$
Threshold	User defined	Calculated	User defined
Max phrase length	No limitation	No limitation	User defined
Flexibility	Only frequency	Any	Any
Amount of information	Low	Medium	High

* m is the max phrase length

4.2 Properties of a Correlation Function

In this section, we describe several key properties of a correlation function. Even though relational probability values are also useful for finding phrases [13], much of the statistical work in building multi-word features focuses on co-occurrence [3,13]. All correlation measures are not equally good at capturing the dependencies between variables. It is because each correlation function has its own bias preferring a set of diagrams to another. Those dependencies can be described in a Venn diagram as shown in Figure 3 – A, B, $A \cap B$, and $A \cup B$.

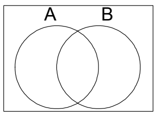

Figure 3. Venn diagram

Piatetsky-Shapiro [12] has proposed three key properties that a good correlation function, F, should satisfy:

P1: if A and B are statistically independent, then F is 0;

P2: F monotonically increases with $P(A,B)$ when $P(A)$ and $P(B)$ remain the same;

P3: if $P(A)$ (or $P(B)$) increases when the rest of the parameters ($P(A,B)$ and $P(B)$ (or $P(A)$)) remain unchanged, then F monotonically decreases.

Tan etc. [16] illustrated those properties and extended to which each of the existing measure satisfies the properties [10]. Some of these properties have been extensively investigated in the data mining literature [8,15,16].

We propose and investigate additional properties. When the universe is fixed, the three events (A, B, and $A \cap B$) can enumerate all possible cases. We list seven possible cases as shown in Table 3. The mark "↑" means increase; "↓" means decrease; "–" means no change. We add two other possible cases – two variables increases with the same ratio. Case 1 has variable A fixed, B fixed, and $A \cap B$ increased. Since this case matches P2 that Piatetsky-Shapiro [12] proposed, we call it P2. We show the proper change of correlation function at the column of "Proper change". Other cases can be interpreted the same way. The proper change of case 1 is increasing [12]. Cases 2 and 3 have the same property name (P3) by similarity – variable A and B could be alternated with each other. The proper change of P3 is decreasing [12].

Table 3. Different cases and their desirable properties

Case	A	B	$A \cap B$	Property name	Proper change
1	–	–	↑	P2	↑
2	↑	–	–	P3	↓
3	–	↑	–		
4	↑	↑	–	S1	↓
5	↑	–	↑	S2	↑
6	–	↑	↑		
7	↑	↑	↑		

Case 4 appears to be the same as the inverse of P2. But not all correlation functions return the same results (eg. Intersection increase for case 1 but no change for case 4). The value of a correlation function should decrease as the value of $A \cup B$ increase when $A \cap B$ remains unchanged in the formula $\frac{A \cap B}{A \cup B}$. Since $A \cup B = A + B - A \cap B$, as A and B increase, the value of $\frac{A \cap B}{A \cup B}$ decreases. Therefore, property S1 is that if $P(A,B)$ increases when $P(A)$ and $P(B)$ remain unchanged, then F monotonically decreases.

Case 5 and 6 also have the same property name (S2) by similarity. The proper change of a correlation value for S2 is monotonically increasing. In the formula of $\frac{A \cap B}{A + B - A \cap B}$, where A and B are bigger than $A \cap B$ and one of A and B are fixed. As B (or A) and $A \cap B$ increases together with the same ratio, the whole value should increase. The property S2 is that if $P(A)$ (or $P(B)$) and $P(A,B)$ increases with the same ratio, then F monotonically increases.

It is hard to tell what desirable change of the correlation function is for case 7 – the values of A, B, and $A \cap B$ increase monotonically together with the same ratio. This property is somewhat related to the frequency/dominant. Too frequent and too rare words are usually removed by some researchers [4,18]. So, we do not determine the proper change for this property.

Another important operation in finding phrases is distinguishing between positive and negative correlations (Property S3). Tan et al. [16] described this property in detail. Since positive correlation is much more important than negative correlation in finding phrases, we only measured the change of correlation values over positive correlation. Statistical independence can be measured by the determinant operator, where Det $(A,B) = A \cap B \times \overline{A} \cap \overline{B} - A \cap \overline{B} \times \overline{A} \cap B$. Thus, a singular diagram D is independent when whose determinant is equal to zero [16]. Measuring their *cross product ratio* (*CPR*) can assess the significance of the correlation between A and B [13] and is defined as:

$$\log CPR(A,B) = \log \frac{P(A,B)P(\overline{A},\overline{B})}{P(\overline{A},B)P(A,\overline{B})} \tag{1}$$

Negative correlation has negative *log CPR* value. Property S3 is that F can distinguish positive and negative correlation of A and B.

Now we examine 15 correlation functions properties P1 through S3. A complete listing of the correlation functions that is examined for further properties in this study is given in Table 5. Table 6 illustrates if the functions satisfy the properties. Each property can have some conditions ($A>B$ or $A<B$). We present the desirable change for each property and conditions. If a correlation function can detect statistical independence of a diagram, we mark "detect independence" in the row of P1. The desirable changes for property P2 though S4 are described above. The mark "positive correlation" for S3 means that the correlation function is able to distinguish positive and negative dependences. We put a check mark "√" when a correlation function satisfies a property otherwise we put "×".

Most of the correlation functions (CE [16], OR [16], YQ [16], YY [16], KP [16], MI-2 [16], PS [16], AEMI [2], MI-3 [13]) satisfy all properties, except CV [16], IT [16], CF [16], AV [16], and MI[2], which do not satisfy S2. Consider $P(A_1)$=0.04, $P(A_2)$=0.4,

$P(B)$=0.5, $P(A_1 \cap B)$=0.03, $P(A_2 \cap B)$=0.3. CV, IT, CF, AV, and MI return the same correlation value, even though their support and *log CPR* values are different. Therefore, satisfying the property S2 is critical for measuring the correlation of a diagram.

4.2.1 Estimating probability values

In order to use a correlation function, we need to estimate the probability values: $P(A)$, $P(B)$, $P(A,B)$. Suppose we have a web page that contains five words:

“`banana ice-cream banana ice-cream spoon`”.

There may be a simple way to calculate the probability of each word and pair: $P(w_1)$=2/5, $P(w_2)$=2/5, and $P(w_3)$=1/5, where w_1= “banana”, w_2= “ice-cream”, w_3= “spoon”, and the total page size is 5. The intersections are $P(w_1,w_2)$=2/4, $P(w_2,w_3)$=1/4, $P(w_1,w_3)$=0, etc, where total possible number of phrases is 4. However, when we calculate $P(w_1, \overline{w_2})$, it causes error by yielding a negative value (-1/10=$P(w_1)$–$P(w_1,w_2)$=2/5-2/4).

We need to calculate for each word the percentage that the word can come before other words and the percentage that the word can come after the other words, called pre-percentage and post-percentage respectively. The pre-percentage of w_1 is 2/(5-1) – w_1 occurred two times, and since we calculate pre-percentage, w_1 may not come to the last position in the page, so that we subtracted one from the total page size 5. The post-percentage of w_1 is (2-1)/(5-1) – even though w_1 occurred two times we have to subtract 1 from 2. It is because w_1 occurred at the first position of the web page; since we calculate post-percentage, w_1 may not come to the first position in the page, so that we subtracted one from the page size 5. We listed the pre- and post-probability for the other word in the Table 4. $P(w_1, \overline{w_2})$ becomes 0 (=1/4–1/4) and the result is very reasonable because “banana” and “ice-cream” are adjacent all the time in the sample web page.

Table 4. Percentage of each word

Word	Percentage		
w_1=“banana”	Pre	$P(w_1, ?\ )$	2/(5-1)=1/4
	Post	$P(\ ?\ , w_1)$	(2-1)/(5-1)=1/4
w_2=“ice-cream”	Pre	$P(w_2, ?\ )$	2/(5-1)=2/4
	Post	$P(\ ?\ , w_2)$	2/(5-1)=2/4
w_3=“spoon”	Pre	$P(w_3, ?\ )$	(1-1)/(5-1)=0
	Post	$P(\ ?\ , w_3)$	1/(5-1)=1/4
“banana ice-cream”	$P(w_1, w_2)$		1/(5-1)=1/4
“ice-cream spoon”	$P(w_2, w_3)$		1/(5-1)=1/4
“banna spoon”	$P(w_1, w_3)$		0

5. EXPERIMENTAL EVALUATION

5.1 Evaluation data and procedures

We use two evaluation data sets: one for phrases and the other for a UIH. To evaluate the effects of correlation functions on phrases generated by VPF, we use a New York Times article about internet companies going public in the stock market [9]. We asked 7 human subjects, other than the authors, to read the article and choose their top 10 meaningful phrases. A total of 45 unique phrases were found by the 7 human subjects.

To evaluate the effect of phrases on a UIH we used the data obtained from our departmental web server. By analyzing the server access log from January to April 1999, we identified hosts that were accessed at least 50 times in the first two months and also in the next two months. We filtered out proxy, crawler, and our computer lab hosts, and identified “single-user” hosts, which are at dormitory rooms and a local company [2]. We chose 13 different users and collected the web pages they visited. The number of words on the web pages visited by each user was on the average 1,918, minimum number of words was 340, and maximum was 3,708. For each page, we generate phrases and add them to the original file.

5.2 Evaluation criteria

First, to evaluate the correlation functions for VPF, for each correlation function, we measure the percentage of phrases found by VPF that match those found by a human subject. We then average the percentage over all 7 human subjects. The higher the percentage, the more phrases found by VPF using the correlation function match the human subjects. This serves as an estimate of how effective each correlation function performs with respect to humans.

Second, to evaluate a UIH, we use both qualitative and quantitative measures. Qualitatively, we examine if the cluster hierarchies are reasonably describing some topics (meaningfulness). Quantitatively, we measure the shape of the cluster trees by calculating the average branching factor [14] (ABF). ABF is defined as the total number of branches of all non-leaf nodes divided by the number of non-leaf nodes.

We categorized meaningfulness as ‘good’, ‘fair,’ or ‘bad’. Since the leaf clusters should have specific meaning and non-leaf clusters are hard to interpret due to their size, we only evaluated the leaf clusters for meaningfulness. This measure is based on interpretability and usability [7] and checks two properties of the leaf: the existence of related words and possibility of combining words. For instance for the related words, consider ‘formal’, ‘compil’, ‘befor’, ‘graphic’, ‘mathemat’, and ‘taken’ are in a cluster, even though ‘befor’ and ‘taken’ do not have any relationship with other words, since other words are classified as a class name, this cluster is evaluated as ‘good’. And for the possibility of combining words, consider ‘research’, ‘activ’, ‘class’, and ‘web’ are in a cluster. In this case the meaning of the cluster can be estimated as ‘research activity’ or ‘research class’[17], so we regard this cluster as good. A cluster is marked as ‘good’ when it has more than 2/5 of the words that are related or has more than 2 possible composite phrases. This is hard to measure, so we tried to be as much skeptical as possible. For example, suppose a cluster has ‘test’, ‘info’, ‘thursdai’, ‘pleas’, ‘cours’, ‘avail’, and ‘appear’. In this case one can say ‘test info’ or ‘cours info’ are possible composite phrases, but ‘test info’ does not have any conceptual meaning in our opinion, so we did not count that phrase. A cluster is marked as ‘other’ when a leaf cluster has more than 15 words because a big leaf cluster is hard to interpret. ‘Fair’ leaf clusters are those that are neither good nor bad.

We categorized shape as ‘thin’, ‘medium,’ or ‘fat’. If a tree’s ABF value is 1, the tree is considered a ‘thin’ tree (marked as ‘T’ in the following tables). If the ABF value of a tree is at least 10, the tree is considered a ‘fat’ tree (marked as ‘F’). The rest are ‘medium’ trees (marked as ‘M’). We consider one more tree type: ‘conceptual’ tree (marked as ‘C’), which subsumes ‘M’ or ‘F’ type trees. A conceptual tree is one that has at least one node with more than two child clusters and more than 80% of the words in each child cluster have similar meaning. Since we prefer a tree that can represent meaningful concepts, 'C' type trees are the most desirable. 'T' type trees are degenerate (imagine each node in the hierarchy has only one child and the hierarchy resembles a list, which is usually not how concepts are organized) and hence undesirable.

Table 5. Correlation functions

#	Correlation function	Formula
1	ϕ-coefficient (CE)	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(a)P(b)(1-P(A))(1-P(B))}}$
2	Odds ratio (OR)	$\frac{P(A,B)P(\overline{A},\overline{B})}{P(A,\overline{B})P(\overline{A},B)}$
3	Yule's Q (YQ)	$\frac{P(A,B)P(\overline{AB})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{AB})+P(A,\overline{B})P(\overline{A},B)}$
4	Yule's Y (YY)	$\frac{\sqrt{P(A,B)P(\overline{AB})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{AB})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}}$
5	Kappa (k) (KP)	$\frac{P(A,B)+P(\overline{A},\overline{B})-P(A)P(B)-P(\overline{A})P(\overline{B})}{1-P(A)P(B)-P(\overline{A})P(\overline{B})}$
6	Mutual Information (MI-2)	$\frac{P(A,B)\log\frac{P(A,B)}{P(A)P(B)}}{\min(-P(A)\log P(A),-P(B)\log P(B))}$
7	Conviction (CV)	$\max(\frac{P(A)P(\overline{B})}{P(A\overline{B})},\frac{P(B)P(\overline{A})}{P(B\overline{A})})$
8	Interest (IT)	$\frac{P(A,B)}{P(A)P(B)}$
9	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
10	Certainty factor (CF)	$\max(\frac{P(B\mid A)-P(B)}{1-P(B)},\frac{P(A\mid B)-P(A)}{1-P(A)})$
11	Added Value (AV)	$\max(P(B\mid A)-P(B),P(A\mid B)-P(A))$
12	Klosgen (KL)	$\sqrt{P(A,B)}\max(P(B\mid A)-P(B),P(A\mid B)-P(A))$
13	MI	$\log\frac{P(A,B)}{P(A)P(B)}$
14	AEMI	$P(A,B)\log\frac{P(A,B)}{P(A)P(B)}-P(\overline{A},B)\log\frac{P(\overline{A},B)}{P(\overline{A})P(B)}-P(A,\overline{B})\log\frac{P(A,\overline{B})}{P(A)P(\overline{B})}$
15	MI-3	$P(A,B)\times\log\frac{P(A,B)}{P(A)P(B)}$

Table 6. Properties of correlation functions

Property	Conditions	Correlation function / Desirable change	IT, MI	CV, CF, AV	CE, OR, YQ, YY, KP, MI-2, PS, KL, AEMI, MI-3
P1		Detect independence	√	√	√
P2	$A=B$	↑	√	√	√
P3	$A<B$	↓	√	√	√
	$A>B$	↓	√	√	√
S1	$A=B$	↓	√	√	√
S2	$A<B$	↑	×	×	√
	$A>B$	↑	×	√	√
S3		Positive correlation	√	√	√

Table 7. Average percentage of matching phrases w.r.t. human subjects

Correlation	CE	OR	YQ	YY	KP	MI-2	CV	IT	PS	CF	AV	KL	MI	AEMI	MI-3
% of match	11	19	16	11	11	11	26	6	25	7	5	18	5	31	26

Table 8. Top 10 phrases found by VPF (best/worst correlation function) and found by 2 humans (closest/farthest) from VPF

Name	Top 10 phrases
AEMI	ventur capitalist, silicon vallei, invest banker, go public, new york, m whitman, morgan stanlei, internet stock, wall street, doesn t
AV	capit menlo park, charl schwab, dlj direct, equal stand, fiercest competitor, i p o, menlo park, net loss, p o, research report, ruth porat, san francisco, wit capit group
H1	silicon valley, wall street, venture capitalist, merrill lynch, feeding frenzy, investment bankers, new york times, web site, I.P.O., first come first served
H2	internet euphoria, thomson financial securities data, silicon valley, trumps any antagonisms, antique fishing decoys, swimming upstream, historical brakes, web frenzy, chock-full of comments, internet innovation

5.3 Results and analysis of correlation functions in VPF

Table 7 lists the average percentage of matching phrases for each correlation function. AEMI matched the most phrases (31%), followed by CV/MI-3 (26%). VPF was significantly affected by the correlation function. Correlation functions that satisfy all the properties were generally better than those that do not (except CV) – IT, CF, AV, MI yielded the lowest 4 values. The reason why CV performed well could be that CV did not use the direct correlation information, $P(A,B)$ or $P(\overline{A}, \overline{B})$; but, it used the indirect/counter correlation information, $P(\overline{A},B)$ or $P(A, \overline{B})$.

Table 8 lists the top 10 phrase found by VPF with the best and worst correlation functions (AEMI and AV) and by the closest and the farthest human (H1 and H2). The average percentage of matching phrases between the human subjects is 20%. This means different people have diverse notions of meaningful phrases. Since AEMI achieves a significantly higher percentage (31%) than the highest percentage shared by the humans (26%), VPF with AEMI was quite effective in identifying phrases.

Table 9. Use words and phrases in a UIH

User	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	Sum
# of L	6	2	13	8	4	5	3	10	8	15	1	6	4	85
Good	6	2	4	5	3	3	3	10	5	7	1	3	4	56
Fair			8	3	1	2			3	8		3		28
Other			1											1
G %	100	100	31	63	75	60	100	100	63	47	100	50	100	**66**
ABF	3.5	2	5	3.4	2.5	3	2	5.5	3.4	3.8	1	3.5	4	
Shape	M	M	M	M	M	M	M	C	M	M	T	M	M	

Table 10: Use only words in a UIH

User	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	Sum
# of L	4	4	3	6	4	4	2	6	4	8	8	4	2	59
Good	3	2	2	6	4	3	2	6	2	1	1	2	2	36
Fair	1	2	1			1			2	7	7	2		23
Other														0
G %	75	50	67	100	100	75	100	100	50	13	13	50	100	**61**
ABF	2.5	2	2	2.7	2	2	2	2.2	2.5	2.4	2.4	2.5	2	
Shape	M	M	M	M	M	M	M	C	M	M	M	M	M	

5.4 Results and analysis of phrases in a UIH

We compared two different data sets: one consists of only words and the other consists of words and phrases – the phrases were collected by VPF using the Average method for thresholds. Table 9 and Table 10 illustrate the results. The results from the data with phrases presented more meaningful leaf clusters (66%) than the results with only words (61%). Tree shapes were similar (medium) in both methods.

6. CONCLUDING REMARKS

To improve a UIH, we proposed using phrases to enrich the vocabulary. Our empirical results indicated adding phrases can generate more meaningful UIH's. To identify phrases, we proposed VPF, which did not have a limitation on the length of phrases. We did not compare VPF with existing phrase finding algorithms empirically because their performance depended on the threshold that were usually determined in an ad hoc way. Our theoretical comparison explained the advantages of our VPF algorithm. Correlation functions affected the behavior of VPF. We also proposed two additional desirable properties (S1 and S2) that characterized an appropriate correlation functions. Generally, correlation functions that satisfied all desirable properties yielded more reasonable results (except CV). VPF with the AEMI correlation function could match the most phrases identified by humans – its results appeared to be even better than human's. Using the phrases (found by VPF) improved a UIH by 5% in terms of interpretability. We did not analyze the differences among the UIH's obtained from the various users because of the large amounts of web pages used in our experiments.

One weakness of VPF is that it sometimes generated phrases that are too long. We can prevent this problem by using more information – the correlation values of all words like Chan's [2]. Our future work is to modify VPF to calculate the correlation values that span beyond adjacent words.

7. REFERENCES

1. Billsus, D., and Pazzani, M.J. A hybrid user model for news story classification, Conf. User Modeling, 1999.
2. Chan, P.K. A non-invasive learning approach to building web user profiles, KDD-99 Workshop on Web Usage Analysis and User Profiling, 7-12, 1999.
3. Chen, L., and Sycara, K. Webmate: A personal agent for browsing and searching, In Proc. 2nd Intl. Conf. Autonomous Agents, pp. 132-139, 1998.
4. Croft, B., Turtle, H., and Lewis, D. The use of phrases and structure queries in information retrieval. In Proc. SIGIR-91, pp. 32-45, 1991.
5. Fagan, J.L. Automatic phrase indexing for document retrieval: an examination of syntactic and non-syntactic methods, Proc. 10th SIGIR, 1987.
6. Frakes, W.B., and Baeza-Yates, R. Information Retrieval: Data Structures and Algorithms, Prentice-Hall, 1992.
7. Han, J. Data Mining Concepts and Techniques, San Francisco : Morgan Kaufmann Publishers, 2001.
8. Hilderman, R. and Hamilton, H. Evaluation of interestingness measures for ranking discovered knowledge. In Proc. Of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining.
9. Holson, L. Feeding a frenzy: Why internet investors are still ravenous. New York Times, June 6 1999.
10. Kamber, M. and Shinghal, R. Evaluating the interestingness of characteristic rules. In Proc. Of the Second Int'l Conference on Knowledge Discovery and Data Mining, Pages 263-266, Portland, Oregon, 1996.
11. Kim, H.R. and Chan, P.K. Learning implicit user interest hierarchy for context in personalization, IUI'03, 2003.
12. Piatetsky-Shapiro, G. Discovery, analysis and presentation of strong rules. In G. Piatetsky-Shapiro and W. Frawley, editors, Knowledge Discovery in Database, pp. 2299-248. MIT Press, Cambridge, MA, 1991.

13. Rosenfeld, R. Adaptive Statistical Language Modeling: A Maximum Entropy Approach, PhD thesis, Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1994.

14. Russell, S., and Norvig, P. Artificial Intelligence A Modern Approach. Prentice Hall, 74, 1995.

15. Tan, P. and Kumar, V. Interestingness measure for association patterns: A perspective*, KDD, 2000.

16. Tan, P. Kumar, V. and Srivastava J. Selecting the right interestingness measure for association patterns. In Porc. ACM SIGKDD, 2002.

17. Zamir, O., and Etzioni, O. Groper: A dynamic clustering interface to web search results, The Eighth International World Wide Web Conference, Toronto, 1999.

18. Zamir, O., and Etzioni, O. Web document clustering: a feasibility demonstration. In Proc. SIGIR-98, 1998.