

Error-Driven Boolean-Logic-Rule-Based Learning for Mining Chat-room Conversations

Tianhao Wu, Faisal M. Khan, Todd A. Fisher, Lori A. Shuler and William M. Pottenger
Computer Science and Engineering, Lehigh University
{tiw2,fmk2,taf2,lase,billp}@cse.lehigh.edu

ABSTRACT

The ephemeral nature of human communication via networks today poses interesting and challenging problems for information technologists. The sheer volume of communication in venues such as email, newsgroups, and chat precludes manual techniques of information management. Currently, no systematic mechanisms exist for accumulating these artifacts of communication in a form that lends itself to the construction of models of semantics [5]. In essence, dynamic techniques of analysis are needed if textual data of this nature is to be effectively mined.

At Lehigh University we are developing a text mining tool for analysis of chat-room conversations. Project goals concentrate on the development of functionality to answer questions such as “What topics are being discussed in a chat-room?”, “Who is discussing which topics?” and “Who is interacting with whom?” The objective is to develop technology that can automatically identify such patterns of interaction in both social and semantic terms.

In this article we present our preliminary findings for a novel technique developed to identify threads of conversation in multi-topic, multi-person chat-rooms. This is the first step towards building models of social and semantic interaction. We term our technique Error-Driven Boolean-Logic-Rule-Based Learning (BLogRBL), a variation on Brill’s Transformation Based Learning [11] [12] [13]. Similar to Brill’s method, rules are automatically derived from templates during learning. It differs from Brill’s technique in that rules take the form of complex expressions of combinational logic. We report on the scope and design of our technique, as well as discussing preliminary results.

1.0 Background And Motivation

The goal of the project is to develop a computational approach towards understanding social and semantic interactions in textual mediums. Chat-room conversation has been identified as the communication medium of interest due to its increasing popularity and the need for research in the area. The 430 million daily instant messages on AOL’s network alone provide a treasure trove of knowledge [3]. Chat conversation is radically different from various other mediums due to its often informal nature. Existing text mining techniques rely on more structured, formal corpuses containing research papers, abstracts, technical reports, etc. Approaches toward understanding the dynamics of chat conversation are limited, and as usage grows the need for automated analysis increases. Due to the dynamic nature of chat conversations, dynamic modeling of social interactions and their contextual topics is a genuine research challenge.

This research is being conducted at the behest of the Intelink intelligence network. Intelink is a secure military communications channel used for critical exchanges of information. Intelink’s goal is to monitor chat conversation over the network and map relationships between users and their topics of conversation to determine the appropriateness of usage and the effectiveness of the communication network. They are interested in information such as the frequency of employee communication, the topics discussed, conversational participants and the focus of the conversations.

This research has, however, applications beyond the scope of Intelink’s needs. In fact, the techniques under development apply to any organization with an internal

communications network, and perhaps to Internet users in general – patrons of chat services such as AOL Instant Messenger (AIM) and IRC (Internet Relay Chat) could benefit from utilizing such a tool.

1.1 Modeling Social & Semantic Relations

The application under development at Lehigh University to model social and semantic interactions is the Social Semantic Builder (SSB). The SSB is a relational modeling tool that utilizes the HDDI™ [6][2][1] text mining infrastructure, and models relationships between distinct conceptual and/or behavioral abstractions. The purpose of the SSB is to determine, analyze, and model the relationships and interactions between these abstract relational entities.

As an example, consider the domain of research papers. Abstractions within this application field include authors of the papers and the concepts they explore. Corpuses are constructed and used to cluster instances of the abstract entities according to their co-relational properties. In this example, two separate models would be created, one of authors who write together and another one of concepts that are similar within the document space.

Continuing the example, the SSB would combine these two models to create a meta-level model between research paper authors and their conceptual content. The meta-model could then be used to analyze and discover previously unknown relationships between authors and their content. This would allow questions such as “What topics are being discussed?”, “Who is discussing which topics?” and “Who is authoring with whom?” to be answered.

Although we have presented the SSB in the context of research article authors and content, social and semantic modeling can take place in any domain involving multiple authors and content.

2.0 Application Domain: Chat

The Social Semantic Builder is a utility with a variety of applications. We have discussed authors and research papers, but there are also students and courses, journalists and newspaper articles, etc. The SSB is designed in such a fashion that its general relational structure could be deployed for mapping between various types of entities. Each application has its own particular issues that need to be addressed, and in this article we address those issues relevant to the analysis of chat conversations.

Some of the questions particular to chat are “Who are the participants in a particular conversation?” “What are they talking about?” “How focused is their conversation?” “How are the participants socially interacting?” “What forms of language do they use to express themselves?”.

A user (such as Intelink) interested in such chat relational models would be able to use the SSB for extracting such information. Chat conversational documents would be input into the SSB, and it would create models of chat participants and conversational topics. A user then could use the models to associate topics with participants, observing which participants discussed a particular topic. Information such as which participants were involved in discussions together, and the topic of those discussions would be readily available. The basic questions of “What topics are being discussed in a chat-room?”, “Who is discussing which topics?” and “Who is interacting with whom?” can thus be answered.

2.1 Chat Input Issues

The SSB accepts input in XML form, utilizing the HDDI™ infrastructure for processing the documents. At present, the HDDI™ System processes the input and then constructs a collection of statistics for analysis. The models for the social and semantic domains are then created and

linked together by the Social Semantic Model Builder.

Some application domains easily map to this input format such as research papers and newsgroup postings¹. The authors are identified at the beginning, and the body of the document or posting can be tagged as content. Chat conversation is not so structured. Chat is often a continuous medium with users entering and leaving a given chat room. Furthermore, even though a chat room may have many users logged in, not all of them may be participating. Of those users who are involved, they do not all participate in the same discussion with all the other users. Often there are several conversations simultaneously taking place between users – a single participant may also be involved in multiple conversations at once. It is an extremely chaotic environment and at first glance seems to lack consistent structure.

In this situation, various chat conversations are interlaced throughout multiple postings and extracting “authors” and their content into single cohesive units for input to the SSB is a daunting task. Furthermore, in our research we have observed that there are numerous categories of chat. Factors such as the number of participants, the topic(s) of chat, the familiarity of users with each other, etc. lead to radically different conversation styles. If a conversation is between acquaintances discussing a common topic, for example, the conversational flow tends to be informal with little attention paid to grammar. If the session is, however, a help session or a discussion medium for a focused topic in which the users don’t know one another, the conversation is typically focused and formal and a broader usage of vocabulary is observed.

Our objective is to partition chat data into collections of postings composed of two or more authors discussing a single topic,

¹ Margaret A. Root defines postings as single messages entered into a network communication system (e.g., chat room or Usenet Newsgroup) [18].

creating input (that we refer to as items) for analysis by the SSB. Thus, each item consists of postings relevant to a single topic, and the users who participated in that topic. Within this framework, the names/screen names of the posters identify authors and the postings identify content. A co-authorship relationship is defined between users based on the content of their postings, and separate content and author clusters are created by the SSB.

Each SSB input item is thus a thread of conversation or discussion revolving around a single, or group of very similar topics.

3.0 Conversational Flow As Threads

As noted, in online chat environments, there are often multiple discussions taking place and within a particular room or channel, authors will participate in multiple discussions, or items [9]. As a result, in the log of a chat session various items overlap as content from multiple discussions is interlaced.

We thus define a thread or item as a collection of multiple authors’ postings grouped together by semantic similarity. Intuitively, a thread should consist of postings that share a similar meaning that is dissimilar from other postings. Clearly, a thread should be composed of the postings of at least two authors.

- | |
|--|
| <ol style="list-style-type: none">1. Mike: What kind of food do you like to eat?2. Anne: I like ice-cream3. Tom: I'm a vegetarian4. Jane: Fruit is my favorite5. Anne: Fruit's cool.6. Anne: I had a banana for breakfast7. Joe: I like ice cream too.8. Mike: My favorite ice-cream flavor is chocolate. |
|--|

Figure 1: A Sample Chat Conversation

In order to be discernable, a thread must contain sufficient semantics to discriminate it from other threads. The challenge lies in

the fact that each posting in a conversation could theoretically represent the start of a thread.

For example, consider the sample chat conversation in Figure 1. At a high level, there is a thread concerning the topic ‘food.’ All eight postings are part of this thread because they are all related to one topic, food. There are however two other threads within this ‘food’ thread. One thread concerns the topic ‘fruit’ (postings 4,5,6) and the other ‘ice-cream’ (postings 2,7,8). Both these ‘sub’ threads fit our intuitive definition of a thread.

This exemplifies a phenomenon we’ve observed within numerous chat samples – the existence of a thread hierarchy. We have observed that the threads in a conversation exist at various levels, with parents and children. Our current research involves the automatic identification of all ‘leaf’ threads in a hierarchy. In future work, we will consider the construction of a model of thread hierarchies.

Note while ‘ice-cream’ and ‘fruit’ were considered leaf threads, ‘vegetarian’ was a topic that did not develop into a thread due to the lack of responses.

3.1 Thread Starts

Determining and extracting threads from an interlaced conversation is a complex task. Our methodology is two-fold. The first step is to classify individual postings as thread starts (the beginning of a thread) or non-thread starts. Once tagged, the second step is to process the non-thread starts, and attach them to their relevant thread starts, thereby creating threads, or items for SSB input.

In what follows we discuss our techniques for identifying thread starts and present experimental validation of our approach.

4.0 Approach

In order to identify thread starts, we’ve developed a novel adaptive learning technique termed Error-Driven-Boolean-Logic-Rule-Based-Learning (BLogRBL).

This approach is similar to Brill’s Transformation-Based Learning, but differs in that BLogRBL combines rules learned using boolean operators.

In the following, we present background information on Transformation-Based Learning, then introduce BlogRBL, followed by a comparison between the approaches.

4.1 Overview Of Transformation-Based Error-Driven Learning

Transformation Based Learning (TBL) is an emerging technique with a variety of potential applications within textual data mining. TBL has been utilized for tasks such as part-of-speech tagging, dialogue act tagging, and sentence boundary disambiguation, to name a few. TBL performs admirably in these tasks since they rely on the contextual information within textual corpora.

The core functionality of TBL is a three-step process composed of an initial state annotator, templates, and a scoring function. The initial state annotator begins by labeling unannotated input (e.g., text) with tags based on simple heuristics. Using a scoring function, the annotated input is then compared to a ‘ground truth’ consisting of the same text with the correct labels. TBL automatically generates transformation rules that rewrite labels in an attempt to reduce the error in the scoring function.

Potential rewrite rules are automatically generated from preexisting human-expert-generated templates. The input in question is then re-annotated using newly generated rules, and once again compared with the ground truth. The procedure selects the best rule (the one with minimal error) and saves it to a final rule sequence. This cycle repeats until the reduction in error reaches a predetermined minimum threshold.

At heart, TBL is a greedy learning algorithm. Within each learning iteration, a large set of different transformation rules can be generated. The rule with the best performance (least error as measured by the scoring function) is chosen. The final set of

rules can be used for classification of new input [11][12][13].

4.2 BLogRBL: Error-Driven Boolean-Logic-Rule-Based Learning

We propose an innovative method that is a variation on the TBL approach described by Brill. The core process is similar: 1) the initial state annotation of text, 2) the usage of templates to generate new rules and 3) the employment of a scoring function for measuring new rule performance. The most important distinction between our method and Brill's, however, is the fashion in which rules are generated. Instead of simply generating rules sequentially, we combine rules using boolean logical operations.

We will discuss the basic framework of our approach and the various templates implemented in experimentation. We applied BLogRBL to the problem of classifying thread starts within complex chat conversations. Ground truths were created with correct labels for each posting, either true indicating a posting is a thread start or false indicating it is not.

4.3 Initial State Annotator

Given the particular application domain, the initial-state annotator tags each chat posting as false (not a thread start) because this was the simplest way to bootstrap BLogRBL.

4.4 Templates

Through manual study of patterns in chat data we developed a number of rule templates. In this article we discuss six of the templates developed – many others have been developed that we do not consider here.

1. A particular word W appears within the current posting, where W is a single English word from a selected word list. This template was chosen since manual inspection of chat data yielded the result that

certain words are often crucial in posting classification. For example a posting with the word “Why” may be asking a question and thus starting a thread, whereas the word “Yeah” indicates a response to a previous posting and is indicative of a non-thread start.

2. A punctuation mark M appears in the current posting, where M is any punctuation mark. Punctuation marks have proven to be valuable in posting classification. For example, a question mark usually indicates a posting is a question, and questions have a high probability of being thread starts.

3. A word with part of speech tag T in the current posting, where T is a part of speech tag from the Brown tag set. Part of speech tags often aid in identifying non-thread starts, for instance interjections such as “yes”, “oh,” and “mmm” often indicate a posting is not a thread start.

4. The current posting's length (the number of words) is C , or the current posting's length is greater than C , where C is a heuristically chosen constant. In this case we observed that thread starts are usually longer postings, as they have to introduce a new topic.

5. The author of the preceding or following posting is also the author of the current posting. Each participant in the chat environment is termed an author. We noted that authors often separate their sentences into several consecutive postings. Thus, it is likely that a posting is not a thread start if the previous posting is by the same author.

6. The timestamp difference between the current posting and the previous posting in seconds. Once again our study of actual chat conversation has shown that if consecutive postings are by the same author, the time difference between them can help identify whether they belong to the same thread. For example, if the time difference between consecutive postings by the same author is short, such as two seconds, it is probably the continuation of a thought and still pertinent to the original thread. On the other hand, often in slower chat conversations, if the time difference between postings is large, the two postings may be unrelated.

Our process instantiates rules based on these templates using features present in the input text; each template generates numerous rules. All of the rules are applied to the text, and within each iteration of the learning process in BLogRBL, the rule with the best performance is chosen.

4.5 Scoring Function

BLogRBL utilizes the F_b -measure (a combined measure of precision and recall) as its scoring function (Equation 1 from [16]). In order to calculate precision and recall for each candidate transformation rule, the templates are applied to chat data and postings are classified as either thread starts or not thread starts, a two-class problem.

Following application of a candidate rule on chat data, the results are compared against the manually annotated ground truth and the rule's precision and recall are computed, yielding F_b . In a given iteration of the learner, the rule resulting in the most significant error reduction is saved.

$$F_b = \frac{(b^2 + 1) PR}{b^2 P + R}$$

Equation 1: F_b measure

BLogRBL employs the F_b -measure as the scoring function rather than the accuracy metric employed by Brill and others due to the large number of true negatives within the chat application domain. We have observed that roughly 10% of the postings in a given conversation are thread starts, and thus true negatives are the 90% balance. As our results in Section 5 show, accuracy is not precise and can in fact be completely misleading – the large number of true negatives often dramatically increases accuracy, while the corresponding number of true positives may be miniscule.

In contrast, the F_b -measure provides a view focused on true positives. For applications such as thread start identification, true positives are of greater interest; the goal is to find as many thread starts as possible. Thus, within this field,

the F_b -measure serves as a more reliable scoring function than accuracy.

4.6 Learning Process Overview

As noted, BLogRBL starts by passing unannotated text through the initial-state annotator and each posting is tagged as a non-thread start. The annotated text is then passed to the learner.

Within the learning stage, BLogRBL generates all possible rules from available templates, and applies them to the annotated text. Logical operations (AND, OR, and NOT) are employed to combine the results of new rules with previously established rules. For example, suppose in the previous learning iteration postings {2,4,5,6} had been identified as thread starts and a newly generated rule identifies postings {1,2,3} as thread starts. Applying different boolean operators, the postings identified as thread starts are as follows: AND yields {2}, OR yields {1,2,3,4,5,6} and NOT yields {4,5,6}.

The scoring function is utilized at this point to compare the current result with the established ground truth. The {logical operator, rule} pair maximizing the scoring function is automatically chosen by the learner.

The learning stage ends when the scoring function doesn't yield improvements and all possible {logical operator, rule} pairs have been exhausted.

The product of BLogRBL is a single rule consisting of several rules combined in a left associated manner using boolean logical operations. Figure 2 depicts an example rule learned by BLogRBL.

```
If (((((((((Len > 5 OR Punctuation = '?') NOT
Len = 1) NOT Len = 3) NOT Len = 2) OR
Punctuation = '"') OR Punctuation = '/') NOT
Punctuation = '(') OR Len > 9) NOT Len = 1)
NOT Len = 3) NOT Punctuation = '=' is TRUE
Then the posting is a thread start.
```

Figure 2: A Rule Learned By BLogRBL

Once a rule is learned, it can be applied to new (unlabeled) data. Following initial-

state annotation, the rule is applied on the data to obtain a set of thread starts.

4.7 Comparison of BLogRBL and Transformation-Based Learning

BLogRBL and TBL make use of the same framework: initial-state annotation, rule generation through templates, and scoring function evaluation. Both methods rewrite text tags within each learning loop and can use the rewritten information within the next loop.

Like Brill’s TBL, BLogRBL can take advantage of templates that depend on rewritten tags. Our current templates listed in section 4.4, however, do not depend on rewritten state. Nonetheless, like TBL, BLogRBL is flexible, and further research may reveal the need for state-dependent templates.

As noted earlier, the main difference between BLogRBL and TBL is the fact that BLogRBL produces a single rule that is a boolean combination of numerous simple rules. TBL on the other hand produces a set of rules that are sequentially applied.

5.0 Results

Chat conversations are often dissimilar, with a wide range of participants, and the conversational topics may be focused or fragmented and are often changing rapidly. In order to obtain preliminary results in validating our methodology, we applied BLogRBL to two different chat conversations with different styles. Due to the classified nature of the Intelink data, however, we have constructed our own test data sets. We describe this process in what follows.

5.1 Test Set Construction

As noted, training and testing was performed on two main samples of chat conversation. The first was an informal conversation on AOL Instant Messenger (AIM) between three researchers; it consisted of 462 postings and is referred as

Dataset 1. There were 3 participants in Dataset 1 and human experts tagged 48 out of 462 postings as thread starts. The second sample was a smaller, focused AIM conversation with a total of 7 participants and is referred to as Dataset 2. This latter sample is more formal and the topic varies less in its 308 postings. Table 1 depicts details of the two datasets. Each of these conversations possesses unique properties and we report results for each. As research progresses, various other styles of conversation will be employed.

Once sample conversational data had been obtained, three human experts manually analyzed it. Each of the two data sets was analyzed, and the three human experts identified thread starts independently. Following this, the three results were correlated to form a ground truth of thread starts for each of the two sample conversations.

5.2 Template Selection

When we first implemented BLogRBL,

	# of postings	# of authors	# of thread starts
Dataset 1	462	3	48
Dataset 2	308	7	40

Table 1: two datasets

we employed a template that permitted any word anywhere in the conversation to be used in learning. This resulted, however, in serious over-training. To solve the problem, we manually selected a small subset of words that were used in the first template. This resulted in a significant reduction in over-training.

The results reported in this article are based on the templates defined in Section 4.4. However, the task of template selection is itself complex [15], and as a result we generated all possible subsets of templates – a kind of ‘template subset selection’. For the six templates in Section 4.4, we generated a total of sixty-three subsets and found that templates 2 and 5 together have the best

performance. Our comparison with decision trees presented later in Section 5.5 is thus based only on templates 2 and 5.

5.3 Results for BLogRBL

Table 2 depicts the results of training on Dataset 1 and testing on Dataset 2. Our best test result is 41.38% for F_{β} with $\beta = 1$. We correctly identify 18 out of 40 thread starts while precision is still reasonable (38.298%). Though modest, these results are promising.

	F-measure (%)	Precision (%)	Recall (%)	TP
Training	22.222	18.841	27.08	13
Test	41.379	38.298	45	18

Table 2: Best test result

It is intriguing that the testing result is even better than training result in table 2. The reason is that the training set, Dataset 1, is more informal than the test set, Dataset 2. Dataset 1 is somewhat chaotic with various degree of focus and topic fluctuation throughout. Moreover, there are many typos and colloquialisms in Dataset 1. All of these factors make it difficult to identify thread starts with a general rule. As a result, the rule learned in training with Dataset 1 doesn't have very good training performance. However, it is still useful for thread start identification in general as shown by its performance during testing. The rule achieves better performance on Dataset 2, which has many thread starts that can be covered by a more general rule. When we use Dataset 2 for training and Dataset 1 for testing (see table 4 below), the training result is better than the testing result. This is a clue that there are more thread starts that can be covered by a general rule in Dataset 2 than in Dataset 1.

5.4 ROC Curve

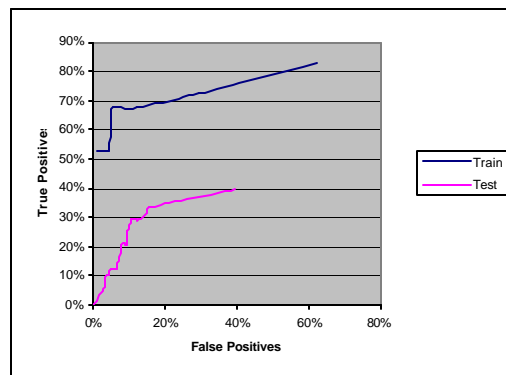


Figure 3: ROC curves for thread start identification

From the ROC curves in Figure 3, we can see that the training result is quite reasonable. BLogRBL gets over 50% of the true positives by sacrificing only about 1% of the false positives. The testing curve starts at (0,0), and increases to (30,11). After that, the curve increases more slowly. Therefore, (30,11) is approximately the best price/performance point – 30% of the true positives are identified at a cost of 11% of the false positives.

5.5 BLogRBL vs. Decision Tree

We applied the decision tree algorithm [17] implemented in [19] to identify thread starts using the same attributes we used in our experiments with BLogRBL (templates 2 and 5 as noted in Section 5.2). The results of this comparison are depicted in tables 3 and 4. Dataset1 was used for training, and Dataset 2 for testing in table 3. In table 4, Dataset 2 was used for training, and Dataset 1 for testing. In this way, we get a clear picture of the performance of the two methods.

Table 3 shows that the decision tree algorithm does not work at all, while BLogRBL works reasonably well. As can be seen, no true positives are found in either training or testing using the decision tree algorithm. On the contrary, BLogRBL identifies 13 of the 48 true positives. Even though the precision (18.841%) is not very high, it is still much better than the decision

tree's precision (0%). Moreover, the test result for BLogRBL is much better than the decision tree's test result. For example, the BLogRBL test result for F_{β} with $\beta=1$ is 41.379%. Compared to the 0% result for the decision tree, BLogRBL's result is excellent.

An interesting point in table 3 is that the decision tree algorithm results in a higher accuracy than BLogRBL even though the decision tree identified none of the true positives. The training accuracy of the decision tree is 89.61%, which is nine percent higher than BLogRBL. Furthermore, the test accuracy for the decision tree is 87.01%, which is also higher than BLogRBL's 83.442%. Clearly the relatively large number of true negatives (90%) is completely skewing the results.

The definition of accuracy is $(TP+TN)/(TP+FP+FN+TN)$, where TP is true positives, TN true negatives, FP false positives, and FN false negatives. The training accuracy of the decision tree in table 3 is near 90% due to the fact that TN is 414, which is much larger than TP (0), FP (0), and FN (48).

We conclude that accuracy as defined above is not a reliable metric for a two-class problem in which one class heavily outnumbers another. Clearly the decision tree algorithm is unsuitable for use in thread start classification, yet it has high accuracy. This is reminiscent of the classic machine learning experiment in which an algorithm achieved remarkably high accuracy in predicting bovine estrus by simply reporting that cows are never in heat!

Clearly further experiments are in order to ascertain the effect of introducing

replication, or weighting, of the thread-start class in the training set. This might mitigate some of the observed effects. Nonetheless, it is worth noting that in comparison BLogRBL performed remarkably well when using the original (unweighted) training set. This may be due to the robustness of the underlying TBL method.

In both tables 3 and 4, the number of true positives is positively correlated with F_{β} . At the same, precision also increases. For example, in table 3, for β equal to 1, F_{β} increases from 22.22% to 41.37% and the number of true positives increases from 13 to 18 while precision increases from 18.84% to 38.29%. Given the data at hand, we conclude that F_{β} is a better evaluation metric for use in thread start identification.

The F_{β} results for BLogRBL are not, however, very high on an absolute scale. Thread start identification is a challenging problem, and further research as discussed in section 7.0 is required.

6.0 Related Work

Understanding social and semantic relationships independent of one another is not a new field. Much significant Social Science research and Information Retrieval research has been conducted in these two fields. We thus do not attempt to survey the literature of social science or IR research here, but rather give a sampling of recent research that is related to our efforts in text mining of chat data. Extracting the information from a chaotic, unstructured forum such as chat to form combined social and semantic models of interaction is however a new area of work.

		F ($\beta=0$.5) %	F ($\beta=1$) %	F ($\beta=2$) %	Accura cy %	P %	R %	TP	FP	FN	TN
Training	BLogRBL	20.0617	22.222	24.904	80.303	18.841	27.08	13	56	35	358
	Decision Tree	0	0	0	89.6104	0	0	0	0	48	414
Test	BLogRBL	39.474	41.379	43.478	83.442	38.298	45	18	29	22	239
	Decision Tree	0	0	0	87.013	0	0	0	0	40	268

Table 3: Dataset 1 used for training, and Dataset 2 for testing

		F ($\beta=0$.5) %	F ($\beta=1$) %	F ($\beta=2$) %	Accura cy %	P %	R %	TP	FP	FN	TN
Training	BLogRBL	52.632	32.653	23.669	89.2857	88.8889	20	8	1	32	267
	Decision Tree	48.611	29.167	20.833	88.961	87.5	17.5	7	1	33	267
Test	BLogRBL	15.625	10	7.353	88.3117	25	6.25	3	9	45	405
	Decision Tree	7.813	3.846	2.551	89.1775	25	2.083	1	3	47	411

Table 4: Dataset 2 used for training, and Dataset 1 for testing

The Galaxy of News [7] project at the MIT Media Lab is an example of early related work in the area of semantic (not social) modeling. It is a system to visualize independent documents and form relationships between them, specifically between news stories. It employed a process to parse the content of news stories and develop weighted relationships between them. This would create a knowledge base of relational networks that could be used to traverse similar news stories, and permit visual interaction with the user. The TDT (Topic Detection and Tracking) research efforts organized by NIST also fall under this category [10].

The Intelligent Network News Reader, HISHO [4] searches through linked news stories for common ones with no need of user input for relevant information. It forms topic clusters of articles, and even keeps track of so called “topic branching articles” for an increased range of semantic connectivity.

The Conversation Map [8] is another product of the MIT Media Laboratory that is more closely related to our work. The system is a Usenet Newsgroup browser that analyzes Usenet postings and creates social and semantic networks of the postings. Due to the inherent threaded nature of Usenet postings, however, social relationships are easily modeled. It provides a graphical interface to track and observe social and semantic relationships from topic to topic.

Upon studying the various factors influencing chat efficiency, The Virtual Worlds Group at Microsoft has developed the Status Client, “a prototype of an interface that shows the status of each user, as determined by keyboard activity” [9]. Through providing more information to users, such as what each member is currently typing, the goal is to simplify the chat environment. A reduction has been observed in the number of posting required to clear out-of-turn and misplaced postings. The issue of chat history loss has been

addressed by a “multi-channel timeline” displaying postings in a real time interface similar to a timeline, with a separate row for each user’s postings. This system, however, is focused on improving chat interfaces rather than on automatic analysis of social and semantic interactions.

Threaded Text [3] is a chat environment developed to address confusion among conversational threads. A chat room is organized into a series of trees, each tree representing a separate thread. Thread boundaries are easily recognizable since they are at the root of each tree. It has led to the observation that some of the fundamental reasons of chat’s chaotic nature is due to the lack of knowledge of “turns-in-progress,” “listening-in-progress” and social and historical context. This effort too, however, is focused on improving chat interfaces using manual techniques. In our survey of the literature we have not come across any work with the goal of automatic modeling of both social and semantic relations within chat data.

Transformation-based learning has been used for dialogue act tagging [14] by Ken Samuel and Sandra Carberry². The authors develop a method to generate dialogue act cues automatically. The cues are used in templates that are applied in transformation-based learning to tag dialogue acts. Some of the templates are similar to the templates used in BLogRBL. For example, Samuel et al. use punctuation marks and the number of words in a dialogue act as templates. Even though dialogue act tagging is a different domain than thread start identification, there are similarities between them, the most obvious being that both of them deal with conversations. We may be able to leverage some of the extensions discussed in [14] to improve the performance of BLogRBL.

Samuel et al. extend transformation-based learning (TBL) in follow-on work reported in [15] in two ways. The first extension is the implementation of a Monte Carlo method to select a subset of possible

rules during learning (rather than trying all rules). This significantly reduces the complexity of TBL. The second extension is the use of boosting as a method to improve the overall performance of the technique. The authors accomplish this by adjusting the scoring function during learning to increase the importance of incorrectly tagged postings. These two extensions have improved the overall performance of transformation-based learning, and such techniques may also be used to improve the performance of BLogRBL.

7.0 Future Work

Our research with chat conversation is currently in its preliminary stages and there is still much to be accomplished. The immediately subsequent step is to continue fine-tuning the templates employed, and the manner they are utilized to improve precision and recall. We continue to seek new templates for identifying thread starts. For example, tags similar to those employed in [14] and [15] may prove to be useful.

Additional chat data samples must be manually tagged in order to provide a greater variety of data sets to train and test on. In truth a variety of tagged chat conversation corpora are necessary. We anticipate that this will stimulate chat conversation research much as the Penn Tree Bank corpora did for probabilistic grammars.

Contextual rules that leverage rewritten labels may be quite useful in this application. Such templates can be implemented in TBL and BLogRBL in order to provide a means for direct comparison between the two approaches. Further manual analysis of chat samples will also result in the development of additional contextual templates.

Various preprocessing techniques for chat data are required. For example, consider the issue of name identification. Depending on the style and format of a chat conversation, participants in a forum may address each other via their screen names or their actual names. Use of screen names

² A dialogue act is similar to our definition of a posting.

facilitates social interactions, as all screen names for a conversation are available. Actual names pose a problem, as they must be (dynamically) mapped to screen names. Actual names and nicknames must be recognized as such within chat conversation and not dismissed as misspelled words.

There are various other chat issues to consider as well such as the idiosyncrasies of chat conversation, using irregular expressions such as brb, u, r, :), etc. The issue of mistyped/misspelled words prevalent throughout chat data due to the informal nature of chat must also be resolved.

As noted earlier, a model for thread hierarchies may also prove essential. Further work on BLogRBL may include considering the improvements on TBL discussed in [15].

8.0 Conclusion

We have developed a framework for modeling social and semantic interactions in chat data. An intuitive definition of a thread has been identified, and BLogRBL, a novel automatic approach for identifying thread starts, has been developed and tested. Results thus far exceed our expectations and serve as motivation for further research.

Our innovative technique, BLogRBL, is a variation on and shares the same framework as Transformation-based Learning (TBL). BLogRBL is unique in that it leverages boolean logic operators to combine selected sets of best rules culminating in a single output rule. In contrast, TBL generates a sequence of rules independent of logical operations.

We have compared BLogRBL with the classic decision tree algorithm. BLogRBL performs admirably versus the decision tree in our preliminary results. Additionally, the F_b evaluation metric has proven to be a better performance metric than accuracy for our application as well as various other two-class problems.

Although much remains to be done, we are confident that extracting and mapping social and semantic relationships in chat and

other venues will prove extremely useful to the field of textual data mining. Certainly the Intelink intelligence network, one of the motivating sponsors of this work, will find this application of genuine value. The potential applications are many, extending into nearly all sectors both industrial and academic. We will continue working toward our overall goal of developing a tool that will automatically discover social and semantic relationships from data expressible in textual form, whether collections of documents, chat data, or other forms of human written communication.

9.0 Acknowledgements

This research was supported in part by the National Science Foundation, Grant EIA-0070457 in the Division of Experimental & Integrative Activities in conjunction with the Intelink intelligence network. Many thanks to Program Directors Rick Adrion and Larry Brandt for their assistance.

We also wish to acknowledge the faculty, students and staff in the Lehigh University CSE Department for their assistance in completing this work. In addition, we especially thank Dr. Alaina Kanfer and M. Cameron Jones for their early contributions to this effort.

Co-authors William M. Pottenger and Tianhao Wu would like to express their gratitude to their Lord and Savior, Yeshua the Messiah, for His continual guidance and support in their lives.

References

- [1] Bader, R., M., Callahan, D. Grim, J. Krause, N. Miller and W. M. Pottenger. The Role of the HDDITM Collection Builder in Hierarchical Distributed Dynamic Indexing. *Proceedings of the Textmine '01 Workshop, First SIAM International Conference on Data Mining*. April 2001.
- [2] Bouskila, F.D. and William M. Pottenger. The Role of Semantic Locality in Hierarchical Distributed. Dynamic Indexing.

Proceedings of the International Conference on Artificial Intelligence (IC-AI'2000), Las Vegas, NV, June.

[3] B. Burkhalter, J. J. Cadiz and M. Smith. Conversation Trees and Threaded Chats. In the *Proceedings of the CSCW'00 Conference*. December 2-6, 2000, 97-105, Philadelphia, PA

[4] Ozaku, Hiromi, Kiyotaka Uchimoto, Masaki Murata, Hitoshi Isahara. Topic Search for Intelligent Network News Reader HISHO. *Proceedings of the 2000 ACM symposium on Applied computing 2000*. Como, Italy, 2000.

[5] Pottenger, William M., Miranda R. Callahan, Michael A. Padgett. Distributed Information Management. *Annual Review of Information Science and Technology* (ARIST Volume 35), 2001.

[6] Pottenger, William M., Yong-Bin Kim and Daryl D. Meling. HDDITM: Hierarchical Distributed Dynamic Indexing. In *Data Mining for Scientific and Engineering Applications*, Robert Grossman, Chandrika Kamath, Vipin Kumar and Raju Namburu, Eds., Kluwer Academic Publishers, July 2001.

[7] Rennison, Earl. Galaxy of News: An Approach to Visualizing and Understanding Expansive News Landscapes. *Proceedings of the ACM symposium on User interface software and technology*. Marina del Rey, California, United States, 1994.

[8] Sack, Warren. Conversation Map: A Content-Based Usenet Newsgroup Browser. *Proceedings of the 2000 international conference on Intelligent user interfaces*. 233 –240. New Orleans, Louisiana, 2000.

[9] Vronay D., Smith M., and Drucker, S. Alternative Interfaces for Chat. *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology (UIST 99)*, 19-26.

[10] www.nist.gov/speech/tests/tdt/

[11] Brill, Eric. Transformation-Based Error Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging. *Computational Linguistics* **21** 94): 543-566.1995.

[12] Brill, Eric. Unsupervised Learning of Disambiguation Rules for Part of Speech Tagging. In *Proceedings of the Very Large Corpora Workshop 1995*.

[13] Brill, Eric. A report of recent progress in Transformation-based Error-driven Learning, *Proceedings of the ARPA Workshop on Human Language Technology.1994*.

[14] Samuel, Ken, Sandra Carberry and K. Vijay-Shanker. Dialogue Act Tagging with Transformation-Based Learning. *Proceedings of COLING/ACL'98*, pp. 1150-1156. 1998.

[15] Samuel, Ken, Sandra Carberry, and K. Vijay-Shanker. "An Investigation of Transformation-Based Learning in Discourse. *Proceedings of the International Conference on Machine Learning (ICML)* , pp. 497-505, 1998

[16] C.J. van Rijsbergen. "Information Retrieval". *Butterworths, London*. 1979.

[17] Quinlan, J. R. Induction of decision trees. *Machine Learning*, 1:81-106. 1986.

[18] Margaret A. Root. Internet/Web Glossary of Terms (W.C.P.L.) http://www.webpilotexplorer.com/WPE_Glossary2.html

[19] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco, CA.