Detection of Topic Change in IRC Chat Logs

Alan P. Schmidt
Department of Computer Science
University of Colorado
Boulder, CO 80310
schmidap@colorado.edu

Trevor K. M. Stone
Department of Computer Science
University of Colorado
Boulder, CO 80310
tstone@colorado.edu

http://www.trevorstone.org/chatsegmentation/

Abstract

We attack the problem of topic segmentation in the domain of Internet Relay Chat logs. In this process, we examine the previous work in text segmentation using a variety of methods. After considering the pros and cons of the methods, we employ Text Tiling, pause detection, and latent semantic analysis because they did not require the usage of large pre-tagged corpora. With these systems in place, we consider the properties and problems that exist when considering the domain of internet chat. To this end, we examine our results and show them to be fair at best.

1 Introduction

The widespread use of the Internet has significantly impacted the language people use to communicate. One of the clearest indications of this phenomenon are chat rooms. The most established chat room system is Internet Relay Chat (IRC). IRC allows users to create and join "channels," which may have an intended topic of discussion or a consistent group of participants. Users can broadcast small amounts of text (from one to around 250 characters) to other channel participants.

Chat room logs offer potentially valuable information. For instance, a system could search for and present to a user conversations about solutions to the user's problem. However, extracting this information requires overcoming several challenges, notably determining conversation boundaries. Improper segmentation could lead to incomplete conversations or conversations which are difficult to follow because of irrelevant interspersed text.

Many researchers have described approaches to segmenting text, but most of them have been used exclusively on formal expository text, which differs significantly from Internet chat. We therefore investigated methods for segmenting chat room logs. We tested the performance of two algorithms – Text Tiling and Latent Semantic Analysis – against a baseline of pause-based segmentation as well as a smaller hand-tagged boundary set.

In this paper, we describe the methods that have been used in the past to perform text segmentation on expository text. With promising methods identified, we have created a system based on TextTiling to attempt topic boundary detection in IRC logs. We continue by discussing what we have identified as the many issues inherent with IRC logs that make this problem more difficult. This discussion leads to our results that show that IRC logs are not as easy to segment as expository text. With this, we discuss areas of future improvement and suggest other methods to try.

2 Previous Work

Several methods have been proposed and evaluated for segmenting text. While most of these have been applied primarily to expository text, chat room text may be similar enough for some of the approaches to perform well in this new context.

2.1 Decision Trees

Work done by Littman and Passonneau(Litman, 1995) presents the idea of breaking text into prosodic phrases. Each of the phrases is then broken down into sets of features based on the linguistic properties of the phrases. Their study tried both hand tuning and machine learning with decision trees to test the features for boundary possibilities. While their results did indeed show that linguistic features seem to relate to discourse structure, the relation of their expository text to IRC logs is not very high. The structure of IRC logs is far more variable and somewhat random.

2.2 Exponential Models

A method proposed by Beeferman, Berger, and Lefferty (Beeferman 1997) suggests a strict statistical modeling that takes advantage of both short- and long-range language models. Their approach involves generating tri-gram models of the words in the corpus that can be used as a short-range estimate of the words that should occur in the current topic. A long-range method boosts the probabilities of seeing certain words locally. By using these two together, partitions can be detected by looking at when long-range probabilities have a dip in performance in comparison with the short-range models. This method relies very heavily on a large training corpus, that the text follow at least some common rules, and that words be used frequently.

2.3 Vector Space Representations

Latent Semantic Analysis (LSA), as described by Landauer, Foltz, and Laham(1998), “is a fully automatic mathematical/statistical technique for extracting and inferring relations of expected contextual usage of words in passages of discourse.” LSA constructs a “semantic space” from a corpus and texts may be compared within this space. LSA has been shown to produce good assessments of text coherence and has been used for document indexing.

Kolenda, Hansen, and Larsen(2000) used Independent Component Analysis (ICA), another vector space approach, to segment 4900 lines of chat from the #CNN news chat channel. Their system identified each 300-character window as fitting one of four recurring topics or the “reject group” of windows which were highly correlated with multiple topics. These researchers did not present their results in a form with which we could easily compare. They gave little indication of how their segmentation compared to human judgment.

2.4 Strict Sentance Overlap

Work cited by Hearst(1994) done by Skorochod’ko(1972) examines what can be learned about a document by comparing documents on the sentence level. By comparing sentences to other sentences for word overlap, an idea of which sentences are connected can be obtained. Detailed examination of these results can help to suggest which discourse model the text is following. By seeing large chunks of localized text that overlap highly, it suggests a possible conversation in that region. Unfortunately, many sentences in chat dialogs may consist of one-word answers to questions, having little meaningful overlap.

2.5 Lexical Cohesion

In research conducted by Kozima(1993), the idea of lexical cohesion profiles was devised. This profile was really a way to describe how certain words are related to other words by way of which words they similar hold meaning to. The idea was to see how words relating from sentence to sentence would hold meaning together based on calculated coherence values. When a sentence was in a lul of cohesion, it meant that a boundary had probably been reached.

2.6 Text Tiling

Text tiling is a method studied by Hearst(1993,1994) which attempts to segment text into coherent discourse units. This method utilizes the types of things learned from the work with sentence overlap and lexical cohesion. As this is one of the methods we chose to base our work on, it will be discussed in the sections to follow.

3 Design and Implementation

From the work that had previously been done in this area, we decided to try a number of different approaches. The methods we tried involved determining where words were in coherent chunks, where the time between messages was great, and determining where semantic relations shift by using latent semantic analysis.

3.1 Text Tiling

Of the methods that have previously been researched, Text Tiling seemed to be a good candidate. Text Tiling divides the text into coherent discourse units that tend to relate to be related by topic. These units are determined by breaking the text into fixed sized blocks and determining the relation between the two at every boundary based on local word context. Points of lower similarity are used as indications of topic boundary.

3.1.1 Turning Logs into Segments

In order to use the text tiling algorithm, text must first be in groupings of segments. To do this, the text from the chat logs was extracted with timestamp, nickname, and server messages removed. From this, all punctuation was removed and capitalization lowercased.

To load the segments, it was important to have a list of stop words that were not used for computation or segmenting. This list was created by counting the frequency of all the words in the corpus and by leaving how many to use as a stop list as a parameter.

Segments were loaded by going through the text line by line, word by word taking the non-stop-list words and adding them to segments. The size of a segment is another parameter that is passed in while executing the program (although was typically a number around 20). The entire log was parsed to create this list of segments.

3.1.2 Segments That Make Up Blocks

In order to do valid comparison, it is done across groupings of segments known as blocks. Once the segments have been created, the blocks are created to refer to some number of segments. The number of segments per block is another parameter (with typical value of 6). With these blocks in place, the number of segment boundaries can be seen and calculated easily. It is important to note that only segment boundaries that can have a full block on either side should be compared.

3.1.3 Calculation of Similarity

For each of the segment boundaries, a calculation is done to determine the similarity between the block on either side. The similarity calculation is that of a cosine measure between the two blocks.

$$sim(b_1, b_2) = \frac{\sum_t w_{t,b1} * w_{t,b_2}}{\sqrt{\sum_t w_{t,b_1}^2 * \sum_t w_{t,b_2}^2}}$$

Where t can be described as the words that occur in the document, and $w_{t,bi}$ is the weight that word t has within the context of block i. This weight was calculated as the word's frequency within the block.

Looking at the equation, its easy to see that the only words that add any score whatsoever are the words that the two blocks have in common. The weights are different between blocks with the same words, yet in different context. This helps for isolation of words that happen frequent in some localized setting.

3.1.4 Smoothing of Similarity Scores

The similarity scores can be quite jagged when graphed, so in order to get something useful out of the data, it must be smoothed. Two types of data smoothing are perfomed on the data.

discrete convolusion

This is a smoothing technique that works very well to smooth large sets of very sporadic data. Two functions are taken and convoluted to produce a third. We can call $x(k)$ the data we wish to smooth, $h(k)$ the convolution function, and $y(n)$ the resultant function. The general procedure of convolusion is:

$$y(n) = \sum_{m}^{N} x(m) * h(n-m)$$

The convolusion function used to smooth the similarity data was suggested by Hearst(1993) to be the following:

$$h(k) = \begin{cases} \frac{1}{bk^2}(bk - |k|) & |k| \leq bk - 1 \\ 0 & \text{otherwise} \end{cases}$$

Where bk is the number of segments per block. This function smooths the data out quite substantially.

median smoothing

The remaining smoothing is done to account for local minima and maxima that we do not necessarily desire in our data. This is done with median smoothing. For every value of data $y(i)$, the median value of $y(i-1), y(i),$ and $y(i+1)$ is chosen and used. For this application, the window size of three was used, however it could be expanded to any number. The resulting data set is incredibly smooth with well defined peaks and valleys.

3.1.5 Depth Scores to Determine Boundaries

With the data ready to use, scores of depths are taken. Since the boundaries are going to happen in areas where the gap has blocks with little similarity that happens in proximity to large areas that do have similarity, we need to know how low the similarity is in comparison with it's neighboring gaps. To calculate this "depth score" for point i, we look right and left to find the two nearest peaks. The similarity value of each is examined and the difference between each and the similarity at i is taken and summed. Boundaries are assumed to be at points when the depth value is very high. This value indicates that not only is the text at i not similar to whats on either side of it, but that the text that is on either side of it corresponds highly with themselves.

3.2 Latent Semantic Analysis

We tested the ability of Latent Semantic Analysis to segment the our test set. We tested LSA using windows of roughly 300 words sliding forward approximately 75 words as well as 200 word windows sliding 100 words. In both cases, the window was extended to the end of the line. We compared pairs of sequential (overlapping) windows to each other in the semantic space provided on the LSA website[1] intended to model general reading through available to a beginning college student. We thresheld the similarity values, marking a boundary where two overlapping windows showed less than 85% similarity.

As indicated previously, many words found in the #l5r corpus would not be present in a typical written training corpus. Some of these words are domain specific, so a semantic space based on texts outside of the Legend of the Five Rings domain will miss many important semantic relations. Similarly, a semantic space trained on Legend of the Five Rings would contain a host of relations of no relevance to any other chat channel. Our main interest is to explore the possibility for segmentation of chat in general, rather than maximizing results on this corpus, so we opted to use the existing general semantic space.

Furthermore, since our corpus is not extensively human-tagged, the LSA learning algorithm would be somewhat impoverished, relying on coarse-grained "texts," such as a full day of chat. Building an Internet chat semantic space from several corpora would probably improve performance by capturing relations among shorthand "words" and prominent words used in casual dialog. We leave this for future work.

3.3 Pauses of Communication

To provide a baseline, we segmented chat logs by pauses. We inserted a topic boundary any time more than one minute passed between chat lines. Unlike the other approaches, the effectiveness of this method will vary between chat channels and even between times of day. It could, however, be used as one of many heuristics in a production system. It also serves as an adequate baseline comparison.

4 Issues and Properties in Internet Chat

When the idea of segmenting topic boundaries in text comes up, it seems to be a pretty straightforward task. However, when we consider this for IRC logs, things are somewhat different. Internet chat logs are corpora that have been little explored, but which offer a potential wealth of data with some interesting properties. As an informal real-time medium, Internet chat shares many traits with speech text. However, text is written, can be read asynchronously, and individual lines may be edited before they are sent, allowing revisions to word choice and a lack of vocalized pauses. Internet chat thus lies between speech text and written discourse. This section points out some of the major differences in IRC logs that make the problem of topic discrimination harder.

[1] http://lsa.colorado.edu/

4.1 Speech Versus Written English

The most noticible difference in IRC logs is that the text is essentially recorded conversation. Conversation itself does not typically follow the same rules as a written language. While written language usually follows the formal definition of the language, dialog tends to be constructed by smaller sentence fragments or clauses. Only when these are strung together can a meaning be infered.

Speech also has the issue of being more than one source of information. In written text, the author is the only one speaking and displays knowledge as clearly and concisely as possible (at least that was what we would like to assume). As soon as two people are involved in a discussion, ideas are more thoroughly exchanged. Points of confusion can be discussed at some length in order to have things make sense.

This is a perfect illumination of why topic discrimination in spoken dialog can be difficult. In written English, meaning is usually contained and layed out within paragraph structures. Spoken English on the other hand uses the mind of a human to maintain and update the current topic. Questions and details all maintaining to the topic at hand can provide a less than concise layout of meaning.

4.2 Changes in Language to Account for Speech

In addition to the general differences between spoken and written English are the problems introduced by using computers to describe spoken English. What has happened is that while we have been exposed more and more to a textual communications device, language has taken on new shapes and styles to provide meaning. Some of the changes are simple conversions of whats used in spoken English. However, there are also a few changes as well. It is also important to keep in mind taht these rules are not necessarily followed by everyone in the same ways.

4.2.1 Capitalization and a Lack Thereof

Conversations do not consist of people saying, “capital I am going to goto capital c-olorado.” Spoken dialog has no indication of case, as the rules of language define where case would be in the event that something needed to be represented in textual form. When people are engaged in chat, many do not take the time to consider where capitalization should occur. The result is text with very little capitalization at all.

The capitalization that is there has it’s meaning altered somewhat. While it is often still used to indicate proper nouns, new meaning has been applied. Conversational speech offers the use of volume to imply importance or to stress the meaning of a certain word to take more precedence overall. With capitalization not being used for much else, using it to imply volume of voice has been accepted (some might argue that using bold text does this, however the laziness of computer users favors using a single holding of shift as apposed to two pressings of ctrl-b).

- These examples should hopefully illustrate this:

- no, this fresh cherry pie is HORRIBLE

- WHAT WERE YOU THINKING?

4.2.2 Indications of Timing

Spoken English is delivered to the recipient word by word as it is being spoken. Chat is usually delievered in sentences (although character by character methods were initially around, line by line were favored). While a person is speaking, they can easily pause for thought, uncertainty, and to pretend to give the listener time to soak in what was just said. Besides pauses in speech, human conversation can be slowed down or sped up as indications of frustration. A number of ways have come about to indicate pauses and timing.

Indications of pause are very commonly repetition of non-vocal characters. Typically, a line of periods between words or sentences with the lengh approximately relating to how long the desired pause is (perhaps approximatly equal to the length to the length of time it takes to read them). For pauses that are meant for more of allowing another to speak or to signify uncertainty, each fragment might be seperated by a new line. Timing is usually represented by either repeating the drawn out sound. Fast speaking is usually associated with anger and yelling, so instead words are usually capitalized and sent as quickly as possible.

- wwwhhhhaaaatttt aaarrree you saying?!

- i...am...going...to...blah...
- its like...

4.2.3 Punctuation Changes

Proper English punctuation is almost unheard of in internet chat. This is another problem that seems to stem from the lack of punctuation in verbal communication. The most noticable punction changes are those which seperate thoughts. In written English, marks like commas, periods, semi-colons, and dashes take care of extending thoughts and connecting clauses. As in speech, a stream of chat is oftern just a series of sentence fragments wih no special attempt to aid reader understanding. With the exception of the ocassional comma used, most marks are replaced by a simple pressing of the return key. In the speaker chooses not to do that, it is often the case that a line of periods is used.

- i wonder. if i walk a mile, will my feet hurt?
- i wonder... if i walk a mile, will my feet hurt?
- i wonder if i walk a mile will my feet hurt?

Like capitalization, lack of formal rule following inspires it's usage for other reasons. Verbal communication can use body gestures to imply that the listener already knows what is about to be said or has heard enough to complete the sentence. This is done textually with the usage of periods. Anywhere that a person might signify that the listner should already know, periods finish the sentence. In a sense, this is like how silence is indicated as gestures often fill the air of silence in verbal communication.

- and thats when ...
- ugh ...

Regular sentence ending type punctuation is still used, however it can be changed slightly to have a greater range of meanings. To stress a level of importance on the sentence, using more than one of the specifed marks can be done. The more marks that are used, the more importance it should be viewed with. When using more than one ending punctuating mark, its also common to mix them depending on which emotion the comment is directed at. One of the more common instances of this is the combination of the question and exclamation marks.

- WHAT?!?!?!
- no, we are going...
- hey, you there????????

One last change that can be problematic is that indications of ownership, speaker or, of contractions are not as commonly used. Anything to speed up typing has been done, and thus the use of apostrophe to display possession is rarely used. As apposed to different symbol, nothing is used at all. This is similar with the use of contractions. There is no attempt to show any indication, as the marks are once again simply stripped away. Quoting people is a common thing to do in chat; however many times, no indication of such is used. Often times the quote is either on a new line or is a copy and paste of the line.

- thats ryans coat
- he said the car is blue
- look what bob said -> 12:34 <bob> man... i love when the clock looks cool

4.2.4 Increased Pronoun Usage

Conversations in general have the property that humans have immediate knowledge about the current topic. Because of this, far less direct referencing of the target subject is done. Instead, people tend to rely very heavily on pronouns to describe the topic at hand. This problem is even further exemplified when the history of the conversation is more readily accessible. In this case, the amount of time before a topic reminder is needed can be longer.

- 12:34 <bob> have you seen jakes new haircut??
- 12:45 <sally> yea, he looks like a total stalker with it

4.3 Changes in Language due to Human Interaction

In addition to changes of language to account for the difference between spoken and written English, there are new changes that deal more with humans growing in their new environment.

4.3.1 Subject Specialization

IRC channels are typically identified by a broad subject in which participants are interested. This typically leads to a high frequency of domain-specific terms,. Our corpus is from the #l5r channel on the Undernet IRC network. Most participants in #l5r play a collectible card game based on Japanese mythology. Discussions are therefore full of references to strange card names, skilled players, game terminology, and coined names of deck types. Furthermore, some normal words have special meanings within the context of the game.

4.3.2 Individual Channel Dynamics

In addition to differences in content, different channels vary in writing style, amount of activity, and focus. Writing style – including vocabulary, spelling accuracy, use of abbreviations – differs from person to person as well as between channels.

Some channels are very active at all times of day, other channels consist of long stretches of silence punctuated by occasional conversations. The usefulness of pauses for segmentation is therefore channel-dependent. The amount of activity, and hence of participants paying attention, on a channel also affects the average length in lines of a single conversation. A system which performs well on a variety of channels therefore must be flexible about segment length.

Further, some channels are more focused than others. Channels intended for technical support, for instance, most discussions are about the product in question. #l5r, on the other hand, has a lot of discussions that have nothing to do with Legend of the Five Rings. Such discussions range from politics to books and movies to computer software to the social lives of channel participants. This breadth combines with the depth of specialization mentioned above to challenge segmentation approaches based on either general language models or domain-specific models.

4.3.3 Multiple Authors

Many past applications of topic segmentation have been on texts produced by a single author. IRC conversations, in contrast, have several "authors," sometimes upwards of 20 at once. This adds several challenges to segmentation.

The first consequence of multiple participants is the presence of multiple disjoint conversations. Groups of participants often hold simultaneous disjoint conversations. Ideally, a segmentation system could disentangle multiple conversations. The participant lists of these conversations sometimes intersect, so establishing participant-conversation pairs won't fully solve the problem. The presence of lines with little content makes approaches which disregard participants infeasible, as the snippet below illustrates:

- <PaulB> Syrneth Navigator
- <Yukimi> Which does?
- <PaulB> cancels the Hiding in the Reefs
- <Yukimi> Ah
- <Dandanar> aaron: awesome tournament report.
- <Yukimi> Tacks to do so?
- <Algernon> Dan:Which?
- <@PaulB> kim: yeah

4.3.4 Noise and Lag

Internet chat contains many lines that could be considered "noise." The most obvious example in the #l5r corpus is a "bot" script which outputs a prestored quote, unique to each participant, whenever a person joins the channel. These lines are rarely relevant to the conversation. While the bot's lines could be filtered, human participants often contribute unrelated lines, such as "<Hitaka> Ah well, bedtime. Later all." Whether such comments should be attached to the nearest conversation, separated into their own "conversation," or filtered entirely will likely depend on the context of a system's use.

Related to the issue of noise is the problem of "lag." Due to technical problems at the network layer, text sent by a user on one server sometimes reaches users on another server several minutes later. Similarly, participants sometimes refer back to a past conversation after scrolling the log after a period of activity. Such lines often appear in an entirely different context. Humans can usually determine the context of the remark, but it presents a challenge for a computer system. At a basic level, these lines will appear as noise, but an ideal system would reattach lagged lines at the right point.

4.3.5 Shorthand

Perhaps the largest difference between Internet chat and typical written discourse is the use of shorthand. Shorthand typically takes the form of acronyms, such as “ROFL” for “rolls on the floor laughing,” “brb” for “be right back,” or an abbreviation of person’s nickname, such as “quix” for “quixote.” Further, a lot of meaningful abbreviations appear in #l5r. Card names, deck archetypes, and people are often referred to by their initials. While the expansion of such acronyms is rarely clear without domain expertise, careful study of the corpus often reveals places where someone asked what an acronym stood for. A segmenting algorithm based on semantic relations would need to handle such shorthand for optimal performance.

4.3.6 Amount of Information per Line

While a line of chat text may be somewhat analogous to a sentence in typical written text, many lines in a chat don’t present much for an algorithm to work with. A person might answer a question with a simple “yes” or “no” which bears no lexical similarity to the question asked. The presence of such lines makes an approach based entirely on comparing lines untenable. At the other extreme, a user sometimes sends an extended quote from an external source as a single line. This may disrupt a scheme which examines a window of a fixed number of words.

4.3.7 “Leetspeek”

One of the larger stereotypes of internet chat is the use of “leetspeek”. In general, this is how people refer to the the activity of using all possible characters to create words. The types of things normally done are to replace regular letters with characters that look like or consist of the letter in some form or another. Another change that is sometimes done is replacing letters or parts ofwords with characters that sound similar. Words written like this tend to get the point across, but are often just goofy looking.

- w3r3 @11 l33+ h@x0r5
- this rulz d00dz

4.3.8 Typos

Just as people tend to get caught on their tongue, fingers can sometimes have problems too. In general, it is not that important for words to be spelled entirely correctly for the meaning to get across. Because of this, as people type faster and faster trying to get some point across, they do not worry about small typos that happen. In conversation, we need only know the pronunciation of the words we use. In written English, we also need to know the spellings; however these days spell check takes care of that for us. Written conversation requires spelling, but doesnt have a checker. Because of this words are often misspelled. Stemming from this, many lines of chat are spent in correcting misspellings. This can be done in a number of ways such as perl regex’s, -+ indications, or even a number of attempts to spell it again.

- Hey, lets go to the stoer
- you know whats hard to spell? antidisistablishmentaryizmm
- Hey DAn! s/A/a
- Hey Dam! -m+n

4.3.9 Emoticons

Since text based communication does not give people the benefit of seeing each other’s expressions, ways of transmiting emotion have come about. Rather than just saying things like “i am happy” or “i am sad”, people found ways that were more fun and subtle. Using the characters available, people have come up with a multitude of faces and pictures.

- :)
- :-(
- O:-)
- @-,-’—
- :-x

4.4 Lack of Editing and Imposed Structure

Articles such as in Hearst typically have a limited focus, show an underlying subtopic structure, and are edited to improve the localization of information. These are usually the result of an intention to share particular information. Internet chat, on the other hand, cannot be edited after the fact and rarely has a specific goal, so we should expect topic cohesion of IRC conversations to be lower and the boundaries fuzzier than in texts typically studied in segmentation.

5 Results

We tested the performance of Text Tiling with three parameter sets, Latent Semantic Analysis with two parameter sets, and the single ”inactivity” algorithm against two days worth of hand-marked logs. We also compared the algorithms with each other on 21 days of untagged logs, including the two days covered of tagged logs. For Text Tiling, we varied the number of words per segment and the number of segments per block, keeping the stop-list at 10 words. For LSA, we applied a roughly 300-word window which shifted by about one quarter of its length for each comparison as well as a window of length 200 shifting by half each iteration. In each case, we looked for proposed boundaries within 10 lines either way of the comparator’s boundary markings. 1 shows the results of comparison to the hand-tagged data. The number of boundary guesses by each method is listed along with the precision and recall.

Table 1: Comparison with Hand-Tagged Data

algorithm	boundaries	recall	precision
hand	223		
time	155	0.283	0.406
lsa100	286	0.619	0.483
lsa75	74	0.215	0.649
tt10-10	169	0.462	0.609
tt10-6	282	0.673	0.532
tt20-6	139	0.404	0.647

Tolerance = 10 lines. Boundaries: number of boundaries the algorithm suggested. Algorithms: hand: by hand; time: pauses $>$ 2 minutes; lsa100: LSA with 200-word windows sliding 50%; lsa75: LSA with 300-word windows sliding 25%; tt10-10: Text Tiling with 10 words per segment and 10 segments per block; tt10-6: Text Tiling with 10 words per block and 6 segments per block; tt20-6: Text Tiling with 20 words per segments and 6 segments per block.

As expected, segmenting by pauses didn’t perform very well. Both Latent Semantic Analysis and Text Tiling were able to produce both recall and precision above 60%, but at the expense of the other measure. The high precision of methods which produced fewer boundaries gives us hope that fine tuning of parameters could produce a system with desirable accuracy. We strongly suspect that a semantic space built on chat text will increase LSA’s performance significantly, so we consider precision or recall above 60% quite a success.

Table 2: Correlation Between Algorithms

algorithm	time	lsa100	lsa75
boundaries	1470	3489	841
time		0.354	0.100
lsa100	0.148		0.196
lsa75	0.181	0.797	
tt10-10	0.193	0.697	0.197
tt10-6	0.162	0.642	0.174
tt20-6	0.200	0.720	0.206

Tolerance = 10 lines. Each cell is the row algorithm’s precision on the column’s algorithm and the column’s recall on the row. Boundaries: number of boundaries the algorithm suggested. For algorithm abbreviations, see 1.

Table 3: Correlation Between Algorithms 2

algorithm	tt10-10	tt10-6	tt20-6
boundaries	1987	3317	1579
time	0.259	0.374	0.214
lsa100	0.400	0.615	0.325
lsa75	0.468	0.691	0.386
tt10-10		0.902	0.735
tt10-6	0.539		0.418
tt20-6	0.920	0.872	

Tolerance = 10 lines. Each cell is the row algorithm’s precision on the column’s algorithm and the column’s recall on the row. Boundaries: number of boundaries the algorithm suggested. For algorithm abbreviations, see 1.

The different methods correlate fairly well, as shown in 2 and 3. Different trials of Text Tiling show a high precision and recall for each other. LSA similarly self-correlates; neither of these facts are very surprising. However, the compared predictions of LSA and Text Tiling often match (within the 10-line fuzzy zone), producing at best 72% precision. Of course, in such a case the recall drops significantly. This correlation suggests that there may be unmarked boundaries found by the algorithms but not marked by humans. This isn’t surprising, because we noted while we tagged the hand-marked log file, we spotted places where the topic shifted between similar topics, which we therefore didn't mark as the start of a new conversation. In a dynamic and casual medium like IRC, even experts will be challenged by whether to segment at a particular position. The desired size and granularity also depends on the application using it, so it is possible that the methods presented here are preferable in some situations than the hand-tagged logs.

/

6 Future Work

One major challenge we faced was the difficulty of marking boundaries by hand for any large portion of the corpus. If someone produced a sizable hand-tagged corpus, several statistical models could be used, including a Latent Semantic Analysis semantic space based on chat and Beeferman's exponential technique. A large hand-marked corpus would give a better picture of accuracy as well.

Our paper addressed finding conversation boundaries, but this is just part of the challenge. We have not addressed the problem of disambiguating multiple concurrent conversations. While #l5r typically only has one conversation at a time, segmented text should be further divided into topically distinct concurrent threads in more active channels. We toyed with applying Latent Semantic Analysis to the collection of each individual's lines within an already-located segment, but our initial results were very poor so we didn't pursue the idea any further. However, with training on chat, and possibly on specific chat rooms, this approach might produce decent results, provided individuals don't participate in multiple concurrent conversations. Other dividing approaches, such as integrating semantic similarity with heuristics, may also prove fruitful.

It would also be interesting to continue examination of the properties of chat. Perhaps there are non-English rules that do infact help in determining when topic boundaries are reached. An example of this might be when new people enter into the conversation, or perhaps topic changes correspond to when legitimate grammar is closest to being used. There are many ways this area could be examined for helpers.

Papers by Fragkou(2002) and Kehagias(2002) provide new methods into segmentation of text. One approach uses dynamic programming to perform linear segmentation by minimizing the global segmentation cost. The similar approach in the other paper uses product partition models to turn text segmentation into an optimization problem which can be solved as well by dynamic programming. These papers were not immediatly discovered and warrant further investigation to determine exactly how applicible they are.

7 Conclusion

In this paper, we show the results of straightforwardly applying two typical text segmentation approaches to Internet chat room text. While the results aren't very impressive, they are encouraging about the possibility of improved performance.

We have investigated and identified many of the inherent problems with internet chat logs that make this problem harder for text in that domain. With further study of these problems, and attempts with some of the new dynamic programming techniques, this problem will have a chance to get far better results.

methods for expository text arent that great

chat topics fuzzy.. even humans had problems

References

Doug Beeferman, Adam Berger, and John Lafferty. Text segmentation using exponential models. In Claire Cardie and Ralph Weischedel, editors, *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 35–46. Association for Computational Linguistics, Somerset, New Jersey, 1997.

P. Fragkou, V. Petridis, and Ath. Kehagias. A dynamic programming algorithm for linear text segmentation.

Marti Hearst. Multi-paragraph segmentation of expository text. In *32nd. Annual Meeting of the Association for Computational Linguistics*, pages 9 – 16, New Mexico State University, Las Cruces, New Mexico, 1994.

Marti A. Hearst and Christian Plaunt. Subtopic structuring for full-length document access. In *Research and Development in Information Retrieval*, pages 59–68, 1993.

Ath. Kehagias, A. Nicolaou, V. Petridis, and P. Fragkou. Text segmentation by product partition models and dynamic programming.

Thomas Kolenda, Lars Kai Hansen, and Jan Larsen. Signal detection using ica: Application to chat room topic spotting.

Thomas Kolenda, Lars Kai Hansen, and Jan Larsen. Dynamical components of chat, 2000.

Hideki Kozima. Text segmentation based on similarity between words. In *Meeting of the Association for Computational Linguistics*, pages 286–288, 1993.

T. K. Landauer, P. W. Foltz, and D. Laham. Introduction to latent semantic analysis. discourse processes, 1998.

Diane J. Litman and Rebecca J. Passonneau. Combining multiple knowledge sources for discourse segmentation. In *Meeting of the Association for Computational Linguistics*, pages 108–115, 1995.

E. F. Skorochod'ko. Adaptive method of automatic abstracting and indexing. In C. V. Freiman, editor, *Proceedings of the Informational Processing Congress 71*, pages 1179–1182. North-Holland Publishing Company, 1972.

This paper is sponsored by eleven zeros.
(0) (0) (0) (0) (0) (0) (0) (0) (0) (0) (0)