SCENE: STRUCTURAL CONVERSATION EVOLUTION NETWORK

BY

MARINA GRIGORYEVNA DANILEVSKY

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Adviser:

Professor Jiawei Han

Abstract

It's not just what you say, but it is how you say it. To date, the majority of the Instant Message (IM) analysis and research has focused on the content of the conversation.The main research question has been, 'what do people talk about?' focusing on topic extraction and topic modeling. While content is clearly critical for many real-world applications, we have largely ignored identifying 'how' people communicate. Conversation structure and communication patterns provide deep insight into how conversations evolve, and how the content is shared. Motivated by theoretical work from psychology and linguistics in the area of conversation alignment, we introduce SCENE, an evolution network approach to extract knowledge from a conversation network. We demonstrate the potential of our approach by taking the task of matching conversation partners. We find that SCENE is more successful because, in contrast to existing approaches, SCENE treats a conversation as an evolving, rather than a static document, and focuses on the structural elements of the conversation instead of being tied to the specific content.

To Josh

Acknowledgments

My sincere gratitude to my adviser, Professor Jiawei Han, who has always encouraged me, and helped focus my confusing ideas into a comprehensive project. This work grew out of a joint project with my fiance, and benefitted directly from discussion with my parents, so I am incredibly grateful to my whole family for both their unwavering emotional support, and their concrete contributions to my academic achievements. This material is based upon work supported by a National Science Foundation Graduate Research Fellowship.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

[1]

It's not just what you say, but it is how you say it. To date, the majority of the Instant Message (IM) analysis and research has focused on the content of the conversation.The main research question has been, 'what do people talk about?' focusing on topic extraction and topic modeling. While content is clearly critical for many real-world applications, we have largely ignored identifying 'how' people communicate. Conversation structure and communication patterns provide deep insight into how conversations evolve, and how the content is shared.

We guide our investigation through theoretical work from psychology and linguistics on "Communication Accommodation Theory" or CAT. First formalized in 1971 by Howard Giles (Giles et al., 1973), CAT suggests that during a conversation, both the lexical and structure of our conversations change based on the phrasing, vocabulary, and sentence structure of our conversation partner. By analyzing these changes, researchers can uncover how people interact, why certain interactions fail, and how to improve them in the future. While findings have a wide variety of applications (Tracy and Haspel, 2004; Rouchdy, 2002; McCann and Giles, 2006; Giles et al., 2006), its primary method of analysis is through hand coding. Hand coding involves researchers going through video or written transcripts, and noting different types of behavior and conversational patterns. These hand-coded results can then be fed into "bag of words" software (Pennebaker et al., 2007), which performs frequency based analysis on categories (e.g. topic, affect, phrasing). This process is slow, cumbersome, and the outcome of such an analysis is only surface level.

[1] Sections of this work are reprinted, with permission, from Marina Danilevsky, Joshua Hailpern, and Jiawei Han. SCENE: Structural Conversation Evolution NEtwork. Proc. of 2011 International Conf. on Social Networks Analysis and Mining (ASONAM 2011), Kaohsiung, Taiwan, July 2011, © 2011 IEEE. This material is posted here and in the subsequent chapters with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the University of Illinois at Urbana-Champaign's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this material, you agree to all provisions of the copyright laws protecting it.

We hypothesize that data mining techniques can be applied to logs of CMC conversations to explore and uncover patterns of communication. However, to explore the patterns in depth, and uncover new interactions, we must first show that these techniques can find known phenomena. This paper therefore focuses on the alignment of conversation structure (Pickering and Garrod, 2006). In other words, two people who have a conversation together will have similar structure to one another. To determine if data mining techniques can detect conversation alignment, we will focus on matching conversation partners (the ability of the computer to find a conversation's partner simply by examining the structure of their text).

Because the focus of this work is on conversation structure, rather than content, we situate this work in a set of IM conversations (between strangers) where the topic was pre-determined. We use a chat log dataset from Hailpern et al(Hailpern et al., 2011) which consists of conversations between 16 pairs of people on pre-determined topics, which allows us to explore the structures of the conversation, rather than the content, which is the focus of our approach.

We begin by discussing linguistic conversation analysis, which will guide our approach, in contrast to the current approaches to chat log analysis in the information retrieval and linguistics communities. We introduce our dataset of chat logs, and describe the metrics used to model the 'structure' of a chat conversation. We then introduce SCENE, an evolution network approach to match conversation partners, and compare it to two other approaches to the matching problem: Euclidean distance and k-means clustering. In contrast to existing approaches, SCENE treats a conversation as an evolving, rather than static document, and focuses on the structural elements of the conversation instead of being tied to the specific content. We show the results of a simple experiment on our dataset, and discuss the implications. We conclude by discussing the work's limitations, its significance, and where future work might lead.

CHAPTER 2

Background

To situate this research, we describe the linguistic literature on analyzing conversation structure and existing approaches to analyzing chat logs.

2.1 Linguistic Conversation Analysis

We inform our analysis and methods through a detailed examination of the theoretical underpinning of conversation structure from the literature on linguistic analysis. The specific details of the following linguistic theories will guide our computational analysis and ground our design decisions in well supported literature.

2.1.1 Communication Accommodation Theory

Communication Accommodation Theory (CAT) was developed by Howard Giles (dissertation in 1971, published in 1973) (Giles et al., 1973). In short, CAT attempts to explain the rationale for how and why peopleÕs language change during conversations. This could be changes in content, phrasing, vocabulary, and other speech patterns. The two most noteworthy patterns of change are:

- **Convergence** describes how an individual shifts his speech patterns in various interactions so that they more closely resemble the speech patterns of the conversation partner(s)
- **Divergence** describes how an individual shifts his speech patterns in various interactions so that they differ from the speech patterns of his conversation partner(s)

CAT is a well accepted, well-researched theory that is considered a major socio-psychological theory of language and social interaction (Tracy and Haspel, 2004). Some current avenues of research look across cultures for how language changes and evolves during interactions (Rouchdy, 2002), or how

age affects interaction in business settings (McCann and Giles, 2006), or public interaction with police offers (Giles et al., 2006). Further, CAT suggest that people will increasingly accommodate their dialogue partners, and try to signal empathy, to elicit approval, trust, respect, or cooperation. This behavior may comprised matching word choices, sentence or utterance length, or even information density.

2.1.2 LSM, Alignment Theory and Mutual Adaptation

In addition to CAT, other theoretical approaches to analyzing conversations have been developed. Linguistic style matching (LSM) refers to the degree to which two people in conversation adjust, their own speaking behavior, or style, to match their partnersÕ behavior. Specifically, LSM suggest that two participants engaged in conversation will utilize similar words in conversation to facilitate coordination in the conversation itself (Niederhoffer and Pennebaker, 2002). In other words, Participant 1 will speak to Participant 2 thereby influencing Participant 2 who will, in turn, influence Participant 1. Pickering and Garrod describe the creation and use of these common words as "Alignment Theory." (Pickering and Garrod, 2006) Further, research on Mutual Adaptation expands upon LSM, suggesting that changes in conversation structure often happen automatically or subconsciously. Further, Cappella's research on Mutual Adaptation states that Òmutual adaptation is pervasiveÓ and that this is Òthe essential characteristic of every interpersonal interactionÓ(Cappella, 1997). Therefore, we aim to show that data mining techniques can uncover this central part of interpersonal communication, which to date, has only been detectable by hand-coding.

2.2 Chat Log Analysis though LIWC

One of the most common methods of conversation analysis is through hand-coding. This is an arduous process in which examiners read a transcript of a conversation and mark all observations of certain key phenomena. To ensure reliability, multiple independent examiners read the same transcript, and inter-rater reliability scores are calculated to determine of the observed behavior did, reliably occur.

In 2006, a linguist named Pennebaker developed Linguistic Inquiry and Word Count (LIWC) (Pennebaker et al., 2007). LIWC is a program that uses a bag-of-words approach to measure content (e.g., judgments, personality, topics, etc.) as well as the more low level elements of language (e.g., punctuation, phrasing, etc). Overall LIWC is a hand/hard coded analysis of language across numerous

dimensions. LIWC processes one, or a group of text files, and outputs the results in each of these dimensions, which include general categories (word count, words per sentence, etc.), linguistic categories (percentage of words in the text which are nouns, verbs, etc.), psychological constructs (e.g., affect, cognition), personal concern categories (e.g., work, home), paralinguistic categories (e.g., assents such as "mmhmm"), and punctuation categories (periods, commas, etc.). While LIWC does provide a surface level analysis of a individual conversation, it does not examine the relationship between multiple conversations, rather, it only produces a set of cursory statistics about one specific transcript. Further, all dimensions and aspects of analysis are both hard-coded and hand-coded. Any previously unknown interaction/word/etc. cannot, therefore, be classified.

2.3 Text Mining Approaches

The work that has been done on chat log datasets in the text mining and IR community has been motivated by the desire to automatically detect conversation topics, often through a combination of classification and statistical language modeling (SLM) techniques. Elnahrawy (Elnahrawy, 2002) evaluates several text categorization approaches for the automatic monitoring of conversation topics in chat rooms. Tuulos et al. (Tuulos and Tirri, 2004) aim to solve the same problem by combining topic modeling techniques with a simple social network representing the relations between the users. Dong et al. (Dong et al., 2006) have a similar goal, incorporating the characteristics of chat messages with term-based categorization for topic detection. Wu et al. (Wu et al., 2002) develop a technique to identify threads of conversation threads in multi-topic, multi-person chat-rooms. Handel et al. (Handel and Herbsleb, 2002) present an empirical study of trends in the content of group chats in the workplace. Kucukyilmaz et al. (Kucukyilmaz et al., 2008) take a term-based approach to examine the content and style of a set of chat conversations, along with known user attributes, with the goal of author prediction, framed as a classification problem. Although this paper is a rare departure from the content-only analysis generally seen in this community, the authors treat conversation style as a static attribute of a specific user, and do not allow for the flexibility that people actually exhibit in altering their communication styles.

The common goal of the text mining community in working with chat logs has always been topic detection, and therefore the only consistent approach has been to discover what people chat about, usually by trying to classify conversations into one of several topics. To our knowledge, no work has gone beyond topic detection from content mining.

Using some simple data mining techniques that focus on the structural features of the chat conversations, we can uncover aspects of interpersonal interaction and communication patterns without being limited by hand-coded dictionaries, or variations on topic. In particular, we focus on solving a simple matching problem (with replacement): given two conversations, with two conversation partners participating in each (for a total of 4 people), and a log of each person's side of their conversation (for a total of 4 one-sided logs), can we find each person's conversation partner, regardless of conversation topic?

2.4 Motivation

Most people are unaware of these subtle but important changes to language during interpersonal communication. However, through a detailed examination of these language patterns, we can uncover and quantify these interactions. To date, literature in computer science, information retrieval and data mining has focused on topic and content of conversation. However, there is a wealth of literature in linguistics and psychology that suggest that the manner and conversational patterns of interpersonal communication is just as critical to understand and capture. Even those in the field of linguistics acknowledge that we do not fully understand the conversational patterns of those using computer medicated communication (e.g. IM) (Pickering and Garrod, 2006). But through this understanding, we can directly influence and understand why communication breaks down, and how to improve it.

This work therefore takes a step towards answering how computer systems can uncover these patterns in communication as postulated by CAT. By pre-determining conversation topic, we can focus on the conversation structure and linguistic changes that occur over time as conversation partners attempt to communicate. To validate the theory of this approach, this work seeks to apply data mining techniques to the vast theoretical literature in linguistics and communication patterns. If data mining techniques can uncover known conversational changes (which normally must be coded by hand), they may be able to then uncover new and yet unknown conversational patterns. This work is intended as a first step in the larger process of applying approaches such as SCENE to chat conversation logs.

CHAPTER 3

Dataset, Approach, & Metrics

3.1 Why Focus on Function Words?

When conversation partners engage, they often seek to align their speech. This is central to human communication (Cappella, 1997). One major aspect of that alignment is the synsemantic[1] or grammatical structure of sentences. This grammatical structure is defined by how content words (e.g. nouns, verbs, adjectives, etc.) are linked together by function words.

Content words are generally tasked with communicating the actual meaning of a communication. Function words, meanwhile, give style and structure to the communication. Compare the brusque 'Broke. Send money.' with the polite 'I am broke; will you please send some money?' Both sentences contain the same content words, but all the words which are present in the polite sentence and missing in the abrupt are function words. In other words, content words can roughly be defined as nouns, verbs, adjectives, and adverbs; function words are every other grammatical class of words, such as prepositions, pronouns, auxiliary verbs, conjunctions, grammatical articles or particles (Delahunty and Garvey, 2010).

The English vocabulary contains almost 100,000 English words, but only about 500 function words. Function words are thus about 0.05% of our vocabulary, and yet they make up about 55% of all the words we speak, hear, and read (Niederhoffer and Pennebaker, 2002). They are the 'structural' aspect of communication.

[1] synsemantic(adj): of a word or phrase meaningful only when it occurs in the company of other words

3.2 Metrics

What are the useful 'structure' metrics to help match conversation partners? In other media such as documents, we would be interested in the number of words per sentence. However, chat conversation are peculiar in that people will often eschew punctuation in favor of hitting the 'return' key at the end of each thought, referred to as an 'utterance'. In this way, utterances (each of which is one line of log text) replace sentences, and so we will note the total number of utterances (lines), in each conversation. as well as the average number of words per utterance (line). As discussed, we pay additional attention to function words. We wish to transform the conversation into 'function word space' by ignoring all content words, and note how much of each conversation is made up of function words. Therefore we will be interested in the total number of function words used, the percent of the conversation that consists of function words (calculated as the count of function words over the count of all words), and the relative frequency of each individual function word (calculated as the count of that word over the count of all function words)[2].

To summarize, we will use 5 'structural' metrics:

- **FunctionWordRatio** = #function words / total #words
- **WordsPerLine** = #function words / #lines
- **LineCount** = total #lines ('utterances')
- **FunctionWordCount** = total #function words
- **FunctionWordsVector** = a vector of the relative frequency of each individual function word.

The FunctionWordsVector can be thought of as representing the conversations position in the 'function word space'. Each feature of this space is a function word, and the vector for each conversation is relatively sparse, since in a ten minute conversation each participant used on average only 34 distinct function words.

We now go into more detail regarding the dataset we used.

[2]It is interesting to note that we tried setting these values to (frequency of function word) / (total count of all words), and generally saw no discernible difference in our results.

3.3 Hailpern et. al. Chat Log Dataset

To situate our investigation to detect conversation structure and match conversation partners, we utilized a set of IM logs[3] generated in 2010 by Hailpern et. al. (Hailpern et al., 2011). To generate this data set, 32 participants were anonymously paired together. Each pair would have two conversations. For each conversation, a **topic** for "debate" was given, and each participant was assigned a **position** (pro or against). The two debate topics were:

- The age at which people gain the right to vote should be lowered to 16 years
- Smoking should be banned in all public places

Unlike most chat log datasets, Hailpern's has the unique and useful feature that the conversation topics were pre-determined. This eliminates a large amount of ambiguity regarding the content. This dataset, therefore, allows us to examine whether SCENE can uncover the structure of conversations when there is no chance of gleaning anything useful from the content. Additionally, since the subjects in the Hailpern et al experiment were strangers, debating fairly serious topics, the conversations were less likely to be full of abbreviations and Internet grammar. It was therefore more likely that our analysis would bear fruit.

3.4 Pre-Processing of Data

The chat logs were pre-processed to extract salient features for analysis. To this end, for each side of a conversation, we performed the following pre-processing steps:

1. Cleaned Words - we eliminated all punctuation, and double spaces. The resulting side of the conversation only consists of words
2. Part of Speech - we used openNLP (Foundation, 2010) to tag the parts of speech in each conversation.
3. Function words only - to examine the synsemantic structure of the logs, we replace all non-function words with a placeholder word, 'XXX.' From this simplified version of the conversations, a dictionary of all function words present in dataset was created.

By the end of this pre-processing, each conversation would be broken up into two files (one representing each side of the conversation). The files would be broken up with one message per line. All

[3]The full dataset will be available online in May 2011, or by request from the authors

content words were represented by a 'XXX' and function words were left as is. Figure 3.1 illustrates the process that each log file went through to be mapped into function word spaces, using a portion of a log from an actual experiment conversation. In addition, a dictionary file was created that represents all 224 function words that were encountered across all 32 conversations.

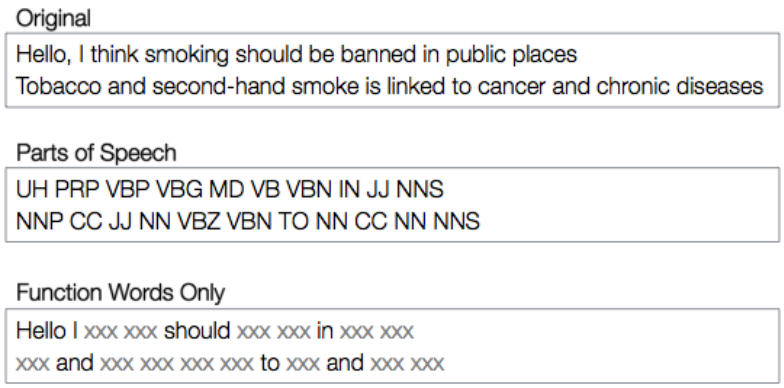

Figure 3.1: *Two sentences, with parts of speech tagged, and with function words identified.*

CHAPTER 4

Analysis & Methods

4.1 Methods

In the 32 conversations we examine, 16 pairs of people each had two conversations with each other, on the assigned topics of banning smoking, or lowering the voting age. In effect, these can be viewed as each pair of people having one conversation, switching topics halfway through. Therefore this dataset consists of 16 conversations, with two participants in each.

Consider a simple matching problem where two conversation pairs are chosen at random. Each person's log of their side of the conversation is examined, and his most likely partner is chosen from the other three people. This is a matching problem with replacement, so although one person may be correctly assigned their conversation partner, the conversation partner in turn might be assigned one of the two non-matching individuals. If choosing randomly, the chance of all four people being matched with their correct conversation partner is $(1/3)^4$ or a little bit over 1%.

Each conversation is represented by a vector of 228 metrics (the Function Words Vector comprises 224 of these, and the other 4 are FunctionWordRatio, WordsPerLine, LineCount, and FunctionWordCount as described above). For every two pairs of conversations, consisting of 4 one-sided conversation logs, we pick up one log at a time, find the nearest log from the other three candidates, and assign it as the likely partner. We perform this matching with every possible pair of conversations from the 16 in our set; therefore we perform (16*15/2) = 120 permutations.

4.1.1 Euclidean Distance

The simplest way to calculate the closest vector among a set of candidates is to use Euclidean distance. Therefore this is our first method. For each of the 120 permutations we calculate the Euclidean distance between all 4 conversation vectors, and for each vector, name its closest neighbor to be the

correct conversation partner. We give each permutation a score of 0 to 4, reflecting the number of correct matches that occurred. On average, using Euclidean distance found correct matches for 1.84 of the 4 conversations.

4.1.2 K-Means Clustering

Another way to match is by clustering. We use k-means with k=2 on each of the 120 permutations. In this case, we need to interpret the results of each clustering more carefully when assigning a score to each permutation. We illustrate with an example: Consider participants A, B, C, and D. A and B are conversation partners, as are C and D. If we cluster these four participants using k-means, we will see one of three outcomes:

1. Cluster1 = {A,B}, Cluster2 = {C,D}. Every conversation looks at its fellow cluster members to find a match. A finds B and earns a score of 1, similarly B finds A, and C and D find each other, all earning a score of 1. Thus the total score for this permutation is 4.
2. Cluster1 = {A,C}, Cluster2 = {B,D} (where C and D might be interchanged, with the same result). Every conversation looks at its fellow cluster members to find a match. A finds C, which is incorrect, and earns a score of 0; B, C, and D fare similarly. Thus the total score for this permutation is 0.
3. Cluster1 = {A,B.C}, Cluster2 = {D} (or any other 3-1 clustering). Every conversation looks at its fellow cluster members to find a match. A must choose between B and C to be its match. A has a 0.5 chance of choosing B, its correct match, and 0.5 chance of choosing C, the wrong match, so the score for A would be best represented as 0.5. B similarly has a 0.5 chance of picking A or C, and also earns a score of 0.5. C has no partner since D is not in Cluster1, and therefore C earns a score of 0. D similarly is marooned by itself, and earns a score of 0. Therefore the total score for this permutation is 1.

The average success rate using k-means over the 120 permutations was finding a match for 1.54 of the 4 conversations. We observed that 23% of the time k-means clustered everyone correctly (Outcome 1), 16% of the time k-means failed completed (Outcome 2), and 61% of the time k-means returned the case with one cluster of size 3, and one cluster of size 1 (Outcome 3).

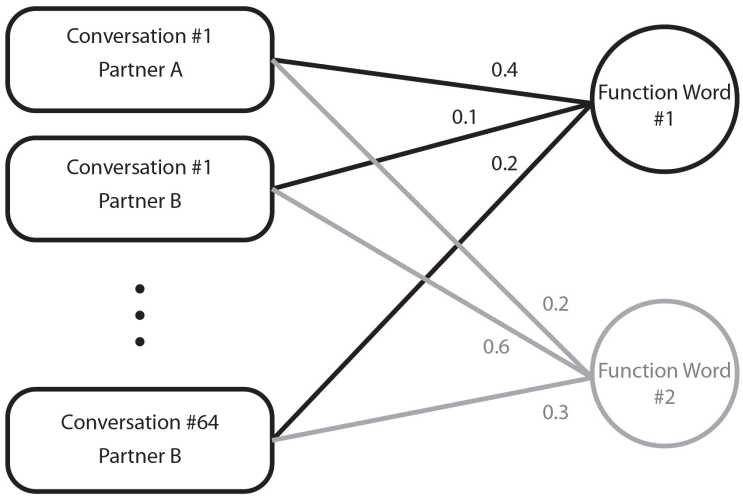

Figure 4.1: *Slice of the dataset visualized as heterogeneous information network. Each rectangle node represents the log of one side of a conversation. Each circle node represents a function word. Each edge weight is the relative frequency of the Function Word in the Conversation.*

4.1.3 SCENE

Both Euclidean distance and clustering with k-means matched fewer than 2 out of 4 conversations correctly on average. We hypothesize this is because these approaches do not take into consideration the evolutionary nature of data, when the data is conversations. We propose a simple alternative approach, a **S**tructural **C**onversation **E**volution **NE**twork (SCENE) which builds on the theories Communication Accommodation Theory and linguistic alignment. SCENE views the conversations as a network, with nodes representing conversations connected to nodes representing function words. The weight on the edges represents the relative frequency of each function word in each conversation. Figure 4.1 illustrates a slice of the network.

An advantage to the network approach that SCENE takes is that it is easy to introduce the concept of evolution over time. It is natural to split a conversation into several parts - for example, when the topic or the nature of a conversation changes. A conversation can therefore be thought of as sequence of conversation parts, each of which is treated as a separate static node, which forms its own edges with the Function Word nodes. This sequence of nodes also forms edges, explicitly mirroring the order of the original conversation. This transformation into sequences allows SCENE to easily examine the conversation as it evolves, rather than deal with each conversation only as a whole, like all current linguistic approaches do (e.g. LIWC). The limited amount of function words, and the sparsity of

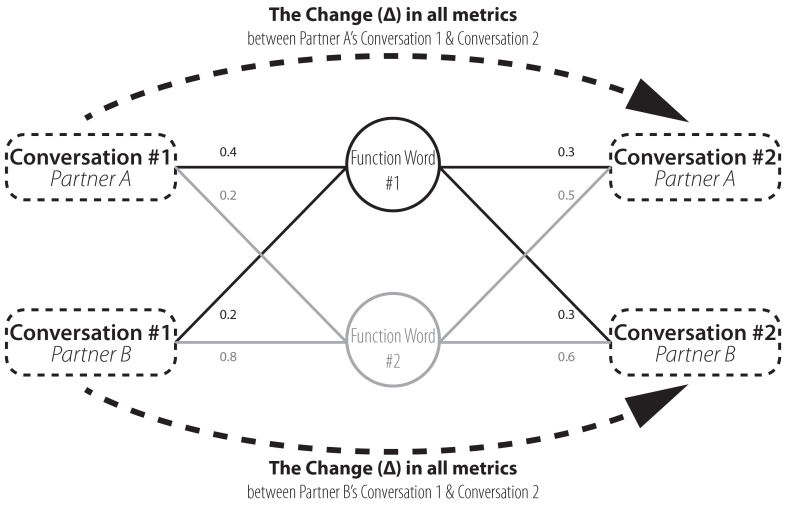

Figure 4.2: *Two conversations, each shown as a sequence of two parts.*
The first halves of the conversations are shown on the left, the second halves are shown on the right.

unique function word use in any given conversation (recall that the participants in our dataset only employed about 34 unique function words on average over two ten-minute conversations), means that transforming a conversation into a sequence of conversations requires very little additional space.

Figure 4.2 illustrates a slice of how two conversations in our dataset might look in the network that SCENE examines. Since all participants participated in essentially one twenty-minute conversation with a change of topic in the middle, it is natural to split each conversation into a sequence of two conversations. It is possible that people change their conversation style when changing topics, because one or both of the participants are more/less interested, or more/less informed on the new topic. However, this approach actually supports our findings even further, that the trend is not based solely on the content of the conversation, since even when users switched topics, they still sought to align with each other. Additionally, it was the only natural splitting point to choose given our dataset - it would have been difficult to support splitting the conversations into, e.g., 3-minute chunks (having no good answer to the immediate question of why 3 minutes would be chosen, rather than 2 or 5).

An advantage of the network framework is that it does not dictate that all conversations must be split into an equal number of sequences - the nature of the data should guide the splitting process. The particular dataset used in this paper creates a simple network structure, with each conversation split into a sequence of two conversation parts, and each participant conversing with exactly one other participant; but we envision this framework to be increasingly useful as longer conversations are added to the network, and participants have additional conversations and change conversation

partners.

Within our dataset, SCENE calculates the difference of each metric for the first and second parts of each conversation, and uses that delta as the representative metric for the conversation as a whole. Therefore the metrics SCENE uses are indicative of the evolution of the conversation, and, as it turns out, are better suited for matching conversation partners. For a conversation consisting of a sequence of Conv1 and Conv2, SCENE calculates the five structural metrics discussed above as follows:

- **FunctionWordRatio, WordsPerLine, LineCount, FunctionWordCount**: each metric is calculated as the difference between the metric for Conv2 and the metric for Conv1 (note that we do not use absolute value, to preserve the trend of the change)
- **FunctionWordsVector** is calculated as the Euclidean distance between the FunctionWordsVector for Conv2 and the FunctionWordsVector for Conv1 (again, in that order, to preserve the trend of the change)

With these five 'delta' structural metrics for each conversation, we again go through each of the 120 permutations, of picking up two random pairs of conversations and trying to find a match for each of the four conversation logs. To calculate the closest conversation from a set of candidates we tried two variations, both driven by the hypothesis that conversations which evolve in a structurally similar manner are a good match:

1. Find the Euclidean distance between the 'delta' structural metrics between the conversation we are trying to match, and every other possible conversation log; choose the logs with the minimum distance as the match
2. For each of the five 'delta' structural metrics of the conversation we are trying to match, order the other possible conversation logs from closest to furthest, resulting in five ordered lists. Employ an ensemble voting method wherein for each list, award the closest conversation 1 point, the second closest two points, and so on. The conversation with the lowest accumulation of points is then chosen as the winner.

The second approach, which gives equal weight to each of the five metrics proved to be the slightly more successful in practice, and it is the one we employ and report.

CHAPTER 5

Results & Discussion

For the matching problem, of pairing four conversation partners together correctly, SCENE proved more adept than Euclidean Distance or K-Means. Although we do not go into a comparison with LIWC[1] or another such similar hand-coded approach, we hold that part of the SCENE paradigm is automatically extracting a significant amount of knowledge from conversations without ever knowing the topic or content of anything that was analyzed.

Table 5.1 presents the matching scores for each algorithm, as discussed in the previous section, illustrating that SCENE is significantly better at matching conversation partners. Table 5.2 goes into additional detail for each algorithm, allowing us to see the distribution of scores which contributed to the mean score for each approach. Of the 120 pairs of conversations examined by each algorithm, we note what percentage of the permutations were awarded a score of 0, 1, 2, 3, or 4, representing the number of conversations whose matches were correctly identified. Recall that because of the difference in calculating the score of the k-means algorithm, it can only award each permutation a score of 0, 1, or 4.

Euclidean Distance does fairly well, although in over a quarter of the cases (28%) it gives a score of 0, by identifying the wrong partner for all four of the conversations. Contrast that SCENE, which completely failed to find any matches in only 6% of the cases, suggesting that it is acting more intelligently and nearly always finding at least one correct match. Additionally, while Euclidean has roughly the same success with finding matches for 0, 2, or 3 conversations, SCENE is by far the most likely to find matches for two of the four conversations. Anecdotal observation of the algorithm suggests that this occurs when one pair of conversations is well aligned, and both partners find each other, whereas the other pair of conversations is badly aligned, and neither partner finds the other. Again, this suggests that there is support for SCENE's approach, but more exploration is needed to better understand which metrics are most informative.

[1] not in the least because the software is not freely available

Table 5.1: *Mean Scores of Matched Conversations*

Approach	Mean Scores
Euclidean	1.84
K-Means	1.54
SCENE	**2.28**

Table 5.2: *Distribution of Matched Conversation Scores*

Approach	0	1	2	3	4
Euclidean	28%	8%	28%	23%	13%
K-Means	16%	61%	n/a	n/a	23%
SCENE	6%	13%	46%	18%	17%

Finally, we notice that k-means clusters 3 conversations together, apart from the fourth conversation, more than 60% of the time. This suggests that straightforward clustering may not be the most productive approach to solving this type of matching problem, because clustering presupposes, in a way, that items mirror each other, and that if A is likely to with B, then B is equally likely to cluster with A. However, in the case of conversation alignment, it would be incorrect to say that person A always aligns with her conversation partner, person B, in the same way and to the same degree that person B in turn aligns with her. The similarity is intuitively too strong a requirement, and we see it proven in practice, as k-means is the worst performing of our compared approaches. It is worth considering that with a larger dataset, consisting of longer conversations between many more than four people, this drawback to the k-means approach may be exacerbated.

5.1 Employing a Decision Tree

The ensemble voting method described at the end of the previous chapter proved fairly successful. However, the decision to give each of the five metrics an equal voice in the voting was simplistic, so in this section we consider yet another approach. Recall the five 'structural' metrics:

- **FunctionWordRatio** = #function words / total #words
- **WordsPerLine** = #function words / #lines
- **LineCount** = total #lines ('utterances')
- **FunctionWordCount** = total #function words
- **FunctionWordsVector** = a vector of the relative frequency of each individual function word.

Table 5.3: *Distribution of Matched Conversation Scores, Individually by Each Metric*

Metric	0	1	2	3	4	Weighted Sum	Sparkline
FunctionWordRatio	43%	14%	28%	9%	5%	1.18%	▆▁▆▁▁
WordsPerLine	23%	8%	28%	15%	26%	2.12%	▆▁▆▁▆
LineCount	14%	18%	16%	33%	19%	2.26%	▁▁▁▆▁
FunctionWordCount	27%	12%	25%	18%	19%	1.91%	▆▁▆▁▆
FunctionWordsVector	6%	20%	46%	17%	12%	2.08%	▁▆▆▁▁

Similar to Table 5.2, we now consider the distribution of matched conversation scores, if each of the five metrics was the only one employed, in turn. Table 5.3 shows the results of this breakdown, and visually illustrates it with the help of sparklines.

5.1.1 VoteTree

The table also gives one possible way to determine the 'success' of each metric, namely, by calculating a weighted sum of the previous columns. For example, the measure for the most successful metric, LineCount, is calculated as $0 * 14\% + 1 * 18\% + 2 * 16\% + 3 * 33\% + 4 * 19\% = 2.26\%$. Thus we can rank the metrics, from most to least accurate:

1. LineCount
2. WordsPerLine
3. FunctionWordsVector
4. FunctionWordCount
5. FunctionWordRatio

We now rerun the matching problem, seeking to pair four conversation partners together correctly for every combination of conversations pairs in our dataset. However, this time instead of employing the ensemble voting method described in the previous chapter, we solicit the top conversation match from each of the five metrics individually, and apply a decision tree algorithm as follows: If the top two metrics, LineCount and WordsPerLine, agree in their choice of the top conversation match, return that conversation. If there is a disagreement, add the top conversation match as chosen by the third metric, FunctionWordsVector, and choose the conversation supported by the majority of the top three metrics. Continue to iterate through the metrics in the order specified above as long as no two metrics agree on a top conversation, until a majority is found. We call this approach *VoteTree*.

Table 5.4: *4-Fold Cross Validation on Two Decision Tree Approaches*

SCENE	VoteTree	FourTree
Cross 1	2.00	2.65
Cross 2	2.67	2.83
Cross 3	3.67	3.67
Cross 4	2.50	2.50
Average	**2.71**	**2.88**

Since we have log data for 16 pairs of people, we use 4-fold cross-validation to avoid overfitting, when testing this algorithm on our data. We use 12 conversations to determine the ranking for the 5 metrics, similar to Table 5.3, and then apply the algorithm to the other 4 conversations to obtain a mean score. Repeat, each time leaving out a different random set of 4 conversations, to obtain a total of 4 mean scores, which we average in turn to get a mean score for conversations matched. The results of each cross validation, and the overall conversation score, are shown in the 'VoteTree' column of Table 5.4

With this ranking, SCENE's mean score for conversations matched increases from 2.28 to **2.71**.[2]

5.1.2 FourTree

A slight variation on this approach is to rank the five metrics based on how often they perform perfectly with a pair of conversations, and match all four conversations correctly, which we consequently refer to as FourTree. If this ranking is to be employed, the metric ranking would be:

1. WordsPerLine
2. LineCount
3. FunctionWordCount
4. FunctionWordsVector
5. FunctionWordRatio

With this ranking, and the same decision tree algorithm applied as above, using the same 4-fold cross-validation, SCENE's mean score for conversations matched increases slightly yet again to **2.88** (see the 'FourTree' column of Table 5.4).

[2]It is interesting to note that in all 4 scenarios of the cross-validation, the metric FunctionWordRatio is always ranked last, and therefore never actually considered. Since our problem is how to pair *four* conversation partners together correctly, for each conversation we have only three choices, and therefore at most need to consult four metrics in our decision tree algorithm to come up with the top conversation to match. This suggests that in the future, this metric would be a good candidate to replace with a different metric.

CHAPTER 6

Conclusion and Future Work

This work is an initial investigation of a very promising topic, and many research directions remain to be explored. The dataset used in our experiments is by no means ideal, since the chat logs from the Hailpern et. al. experiment (Hailpern et al., 2011) were small in both number and length. Two ten-minute conversations is not a long time in which to adjust to another person's speech, and it is encouraging to see alignment occurring in this small time period. The next step is to obtain more chat logs to see if how well our approach generalizes to larger sequences of longer conversations. This would also allow us to build on our analysis which is currently limited to matching up 4 participants, due to the noisiness of the short conversation logs. Additionally, the current dataset is artificially constructed so that every participant has exactly one conversation partner. A more complex dataset where each person has multiple conversation partners would be a logical next step.

There are clearly important inherent differences between synchronous, text-based computer mediated interactions and face-to-face interactions. In face-to-face interactions, the participants are in close proximity to one another while in computer-mediated communication participants are physically distributed from one another. Nonverbal cues are also absent in a text-based computer-mediated setting. Also, the linguistic style of a chat conversation is different from a face-to-face conversation. When chatting over the computer, people tend to be less formal, use more slang and abbreviation, and pay less attention to grammar. It is possible that the fact that the conversation partners were strangers, as well as the 'debate team' nature of the assigned topics (lowering the voting age, and banning smoking) biased the participants to a more formal style of communicating. It would be interesting to test the robustness of our approach if the participants are actively motivated to participate in a the conversation. For example, participants could be assigned a fun, challenging task, such as giving one participant a finished structure, and instructing them to explain to their partner, who is given a collection of raw materials, to create a copy of the structure, all over instant messaging.[1] The promise

[1] This example is inspired by Write It, Do It, an event which is commonly encountered by students who participate in Science Olympiad. A description of the event may be found at http://scioly.org/wiki/Write_It_Do_It

of a financial bonus to the partners who together create the most accurate copy of the given structure, together with the hands-on nature of the task, would likely result in more realistic chat log data.

In this work we introduce SCENE, an evolution network approach to extract knowledge from a conversation network. Our approach is motivated by theoretical work from psychology and linguistics in the area of conversation alignment. We demonstrate the potential of our approach by taking the task of matching conversation partners. We find that SCENE is more successful because, in contrast to existing approaches, SCENE treats a conversation as an evolving, rather than a static document, and focuses on the structural elements of the conversation instead of being tied to the specific content. We further suggest that there is still much work to be done to establish both structural metrics as well as calculated measures used to discover conversation partners, which would be robust across multiple datasets. Although our initial analysis chose metrics and measures in several simple ways, a learning-based approach is definitely indicated for future work.

References

Cappella, J. (1997). The development of theory about automated patterns of face-to-face human interaction. *Developing communication theories*, pages 57–84.

Delahunty, G. P. and Garvey, J. J. (2010). *The English Language: From Sound to Sense*. Perspectives on Writing, Fort Collins, Colorado.

Dong, H., Hui, S. C., and He, Y. (2006). Structural analysis of chat messages for topic detection. *Online Information Review*, 30(5):495–516.

Elnahrawy, E. (2002). Log-based chat room monitoring using text categorization: A comparative study. In *Proceedings of the IASTED international conference on information and knowledge sharing*, St. Thomas: US Virgin Islands.

Foundation, A. S. (2010). Opennlp. *http://incubator.apache.org/opennlp/*.

Giles, H., Fortman, J., Dailey, R., Barker, V., Hajek, C., Anderson, M., and Rule, N. (2006). Communication accommodation: Law enforcement and the public. *Applied interpersonal communication matters: Family, health, and community relations*, pages 241–269.

Giles, H., Taylor, D. M., and Bourhis, R. (1973). Towards a theory of interpersonal accommodation through language: Some canadian data. *Language in Society*, 2(2):pp. 177–192.

Hailpern, J., Danilevsky, M., Harris, A., Karahalios, K., Dell, G., and Hengst, J. (2011). Aces: Promoting empathy towards aphasia through language distortion emulation software. In *Proceedings of the ACM SIG CHI Conference 2011 Conference.*, CHI 2011, Vancouver, BC Canada. ACM.

Handel, M. and Herbsleb, J. D. (2002). What is chat doing in the workplace? In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, CSCW '02, pages 1–10, New York, NY, USA. ACM.

Kucukyilmaz, T., Cambazoglu, B. B., Aykanat, C., and Can, F. (2008). Chat mining: Predicting user and message attributes in computer-mediated communication. *Inf. Process. Manage.*, 44:1448–1466.

McCann, R. and Giles, H. (2006). Communication with people of different ages in the workplace: Thai and American data. *Human communication research*, 32(1):74–108.

Niederhoffer, K. G. and Pennebaker, J. W. (2002). Linguistic style matching in social interaction. *Journal of Language and Social Psychology*, 21(4):337–360.

Pennebaker, J., Booth, R. J., and Francis, M. E. (2007). Linguistic inquiry and word count: Liwc.

Pickering, M. and Garrod, S. (2006). Alignment as the basis for successful communication. *Research on Language & Computation*, 4(2):203–228.

Rouchdy, A. (2002). *Language contact and language conflict in Arabic: Variations on a sociolinguistic theme*. Routledge.

Tracy, K. and Haspel, K. (2004). Language and Social Interaction: Its Institutional Identity, Intellectual Landscape, and Discipline-Shifting Agenda. *Journal of communication*, 54(4):788–816.

Tuulos, V. H. and Tirri, H. (2004). Combining topic models and social networks for chat data mining. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '04, pages 206–213, Washington, DC, USA. IEEE Computer Society.

Wu, T., Khan, F. M., Fisher, T. A., Shuler, L. A., and Pottenger, W. M. (2002). Error-driven boolean-logic-rule-based learning for mining chat-room conversations. In *Lehigh CSC 2002 Technical Reports*, pages 02–008.

Author's Biography

Marina Danilevsky was born in Moscow, Russia, and moved to the north suburbs of Chicago with her family at a young age. She received her Bachelor of Science degree from the University of Chicago in 2007, with a concentration in Mathematics. From 2007 to 2009 Marina worked as a Research Assistant at LECG, an economic consulting group in Chicago. In the fall of 2009 she joined the Department of Computer Science at the University of Illinois, Urbana-Champaign. She plans to continue in the department to earn her PhD in Computer Science with a focus in Data Mining, under Dr. Jiawei Han.