ABSTRACT

HELPING USERS LEARN ABOUT SOCIAL PROCESSES WHILE LEARNING FROM USERS: DEVELOPING A POSITIVE FEEDBACK IN SOCIAL COMPUTING

Venkata Sai Sriram Pillutla, M.S.
Department of Computer Science
Northern Illinois University, 2017
Philippe J. Giabbanelli, Director

Social computing is concerned with the interaction of social behavior and computational systems. From its early days, social computing has had two foci. One was the development of technology and interfaces to support online communities. The other was to use computational techniques to study society and assess the expected impact of policies. This thesis seeks to develop systems for social computing, both in the context of online communities and the study of societal processes, that allow users to learn while in turn learning from users. Communities are approached through the problem of Massive Open Online Courses (MOOCs), via a complementary use of network analysis and text mining. In particular, we show that an efficient system can be designed such that instructors do not need to categorize the interactions of all students to assess their learning experience. This thesis explores the study of societal processes by showing how Text Analytics, Visual Analytics, and Fuzzy Cognitive Map (FCM) can collectively help an analyst to understand complex scenarios such as obesity. Overall, this work had two key limitations. One was in the dataset we used, as it was small and didn't show all possible interactions, and the other is in the scalability of our systems. Future work can include the use of non n-gram features to improve our MOOC system, and the use of graph layouts for our visualization system.

NORTHERN ILLINOIS UNIVERSITY
DE KALB, ILLINOIS

MAY 2017

HELPING USERS LEARN ABOUT SOCIAL PROCESSES WHILE LEARNING FROM USERS: DEVELOPING A POSITIVE FEEDBACK IN SOCIAL COMPUTING

BY

VENKATA SAI SRIRAM PILLUTLA

A DISSERTATION SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

Dissertation Director:
Philippe J. Giabbanelli

ACKNOWLEDGEMENTS

This thesis becomes a reality with the kind support and help of many individuals. I would like to extend my sincere thanks to all of them.

Firstly, I would like to express my sincere gratitude to my advisor, Prof. Philippe J. Giabbanelli. I thank him not only for the continuous support of my Masters study and related research, but also for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my masters study.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Reva Freedman and Prof. Jie Zhou, for their insightful comments, encouragement, and their questions which incented me to widen my research from a different perspective.

Thanks to our collaborator, Dr. Andrew Tawfik, for providing the data to analyze the students' interactions in Chapters 4 and 5 of this thesis.

Finally, I thank my lab mates, Magda Baniukiewicz, Eric Lavin, and Robby Ruschke, for the stimulating discussions, and for all the fun we have had during the course of my masters.

DEDICATION

To my father, for his guidance and support.

To my mother, for her love and affection.

ఒకచో నేలను బవ్వళించు, నొకచో నొప్పారుఁ బూసెజ్జపై,
నొకచో శాకము లారగించు, నొకచో నుత్కృష్టశాల్యోదనం,
బొకచోఁ బొంత ధరించు, నొక్కొక్కతరిన్ యోగ్యాంబర శ్రేణి, లె
క్కకు రానీయఁడు కార్య సాధకుఁడు దుఖ్ఖంబున్ సుఖంబున్ మదిన్..

- భర్తృహరి, సుభాషిత రత్నావళి

"At one place, he sleeps on the floor; at another, he sleeps on a comfortable bed decorated with flowers.
One day, he eats raw vegetables; another day, he may eat a delicious mouthwatering dinner.
At one place, he may wear rags; and, once in a while, he may have the most regal clothing.
A goal-oriented person, an achiever, will not care either the ordeals or the pleasures."

- BHARTRUHARI, *SUBHASHITA RATNAVALI*, 18TH CENTURY

CONTENTS

Chapter Page

Chapter Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

Social computing was defined by Wang et al. to encompass the "computational facilitation of social studies and human social dynamics" [153]. From its early days, social computing has had two foci. One was the development of technology and interfaces to support online communities. The other was to use computational techniques to study society and assess the expected impact of policies [153]. While social computing can be approached using a variety of techniques, the focus of this thesis is to contribute to both aspects of social computing by using text analytics and network science. These two approaches have gained considerable attention over the last decade, and continue to be active fields of research under the umbrella of 'data science'. Text analytics and network science have been widely used to capture rich social interactions through platforms such as Facebook or Twitter. For example, complementary uses of text analytics and network science in Twitter have provided an understanding of the autism community [11], revealed political affinities [99], or highlighted public health concerns [80]. These various achievements have specifically relied on text mining (using the supervised technique known as *classification*), text and/or network visualizations, and network analysis. These different techniques will also be employed in this thesis, and will be further described in chapter 2.

Data scientists have contributed to social computing by frequently asking two questions, in a wide variety of contexts: can we detect what people say about a given topic [65, 98], and can we predict or correlate what they say with what they do in the real-world [99, 80]? The result of these inquiries is frequently an analysis or algorithm that users can provide with specific keywords or time periods of interest. In contrast, less attention has been devoted to the development of systems that help users refine their understanding of a social phenomenon and can in turn learn from users to show a more accurate picture. In this thesis, we seek to develop such systems by using text analytics and network science.

1.1 Contributions

The overarching goal of this thesis is to develop systems for social computing, both in the context of online communities and the study of societal processes, that allow users to learn while in turn learning from the users. In line with Pratt et al., we hypothesize that the creation of such positive feedback can result in more accurate understanding of the complex dynamics that underlie social phenomena [118]. Our overarching goal will be accomplished through two specific aims, thus assessing the benefit of this feedback in both aspects of social computing:

1. Can instructors in Massive Open Online Courses (MOOCs) use an automated system to refine their assessment of the students' learning experience, while refining the system's own assessment?

2. Can a system guide analysts and modelers in exploring data about social processes, while learning from the users how to best guide them?

This thesis is motivated by designing better systems rather than by using a specific tool. Consequently, a variety of techniques will be used. The data in all parts of this thesis consists of text, thus text analytics will always be present. However, text analytics is more a field of research than a specific method. Similarly, network science provides a large collection of methods which are only united in this thesis by representing the input data as a graph/network.

1.2 Outline

The thesis is divided into 3 parts with specific goals. The first, introductory part of this thesis introduces the fundamental notions of text analytics and networks that will be used throughout this thesis (chapter 2), and thoroughly exemplifies their use in social computing (chapter 4). The second part is devoted to aim (1): supporting instructors in assessing students' learning experiences in Massive Open Online Courses. This is accomplished by using a social network analysis in chapter 4, and by applying text classification in chapter 5. The final part of this thesis focuses on aim (2), that is, guiding analysts by providing an interactive visualization environment combining Fuzzy Cognitive Maps, Text Analytics, and Visual Analytics. Chapter 6 explains how an innovative combination of these techniques is designed and implemented as an interactive visualization environment. Finally, chapter 7 concludes this thesis by summarizing our accomplishments regarding our overarching goal, and offers directions for future work.

CHAPTER 2

BACKGROUND: TEXT ANALYTICS AND NETWORK SCIENCE

The main approaches employed in this thesis are text analytics and network science. Their application runs through the different chapters, with at least one of these techniques being core to each chapter, and both being employed either to offer different perspectives on the same problem (Chapters 4 and 5) or in a synergistic manner (Chapter 6). Consequently, this chapter establishes the methodological foundations of these approaches, and offers succinct examples of their applications. Thorough applications are provided in the specific context of each chapter. For example, Chapter 3 provides a complete case study of text summarization and visualization, going beyond the introductory notions presented here. All material related to this chapter can be accessed on a third-party repository at `https://osf.io/amryj`, which includes the dataset I assembled for this chapter, and the software I developed to create network visualizations of text documents.

2.1 Introduction

We start by introducing foundational concepts of network science in section 2.2, using networks formed from text documents as the guiding example. Then, we describe different techniques in text analytics, organized in three intuitive categories. Section 2.3 covers *fully automatic* techniques in which there is no human intervention: algorithms use the text as sole input. In section 2.4, we briefly review *semi-automatic* techniques in which parameters are required in addition to the text, and the user typically has to experiment with different parameter values to get the desired result. This includes word trees and text summarization. Finally, we discuss *supervised* techniques in which the text itself must be annotated by humans; such techniques may also include parameters. Here, such techniques are represented by text classification. We note that none of these sections seeks to provide an exhaustive overview of text analytics, which is beyond the scope of the thesis and the subject of numerous books. Rather, we provide sufficient background to understand the techniques selected in this thesis, and the rationale for their selection. While our focus is on *analyzing* text, the *generation* of text combines many of the text and network concepts developed here. Consequently, we provide it as a supplementary example in the appendix.

The guiding example for most of the algorithms shown in this section is a corpus that we assembled on the Zika virus. The corpus consists of news reports over a period of 10 weeks (28 January 2016 to 11 April 2016), at a time where the Zika virus was commonly discussed. We used Lexis Nexis as news aggregator, which is commonly used as a first step to assemble a corpus [51]. Note that this procedure is skewed towards large newspapers, and could be complemented by other online databases such as Access World News (`http://infoweb.newsbank.com`). Since our corpus was assembled to exemplify the algorithms rather than as a subject of study itself, we did not complement it by other databases to limit bias. Selection of news articles consisted of applying the following criteria in sequence:

(i) find articles that include the keyword 'Zika virus' and 'America' or 'USA' and in our selected time period,

(ii) retain articles for newspapers having at least 6 publications.

Applying (i) resulted in 992 articles. After application of (ii), there were 487 articles, from 25 newspapers. Meta-data about each article was kept in a separate excel file and contained the article's title, author(s), publication date, and the newspaper. The corpus of 487 articles and the meta-data (for all 992 articles) can be accessed online at `https://osf.io/amryj`

2.2 Network Science applied to text

Definitions 2.2.1 and 2.2.2 formally define the notion of graph, and two sub-types of graphs. These are standard definitions in graph theory, and can be found in textbooks such as [29] (pp. 2–4). A graph being a mathematical structure, it does not include a spatial embedding: that is, the same one graph may be displayed in a number of ways depending on the number of dimensions one wishes to see (e.g., 2D, 3D) and how elements should be positioned in these dimensions. To assign a spatial embedding to a graph, we use a graph layout, formally defined in 2.2.3. In this thesis, we will only be concerned with layouts to embed a graph in the plane by assigning a position to the graph's nodes (thus implicitly defining the location of edges).

Definition 2.2.1. *A* ***graph*** $G = (V, E)$ *is composed of a set of vertices (or nodes) V and edges (or links) E. Nodes correspond to entities (e.g., words, persons) and edges correspond to relationships (e.g., words appearing in a same sentence, persons working together).*

Definition 2.2.2. *A graph is* ***directed*** *when its edges have a direction. For example, an edge* $(u, v) \in E$*,* $u \in V$*,* $v \in V$ *indicates a relationship going from u to v. Conversely, a graph is* ***undirected*** *if all edges are bidirectional: an edge* $(u, v) \in E$*,*$u \in V$*,* $v \in V$ *indicates a relationship between u and v.*

Definition 2.2.3. *Given a graph G=(V,E), a* ***layout*** *embeds G in a space* $\Re^n$ *of* n*-dimensions by assigning a coordinate vector* $(v_1, \cdots, v_n)$ *for all nodes* $v \in V$*. When three coordinates x,y,z are used then we have a* $3 - d$ *layout, and when two coordinates x, y are used then we have a* $2 - d$ *layout.*

We can now illustrate these three definitions using text. To do so, we need to state (i) what elements of the text are represented by nodes, (ii) what relationships are captured by edges, and (iii) which specific layout is used. There are many possible choices for each of these steps. In this example, the nodes correspond to words from the text. An edge between two nodes represents the co-occurence frequency (i.e. *bi-gram frequency*) of the associated words, that is, the number of times the words co-occur in the entire text document. Finally, we use the Spring layout. The result is provided in figure 2.1, rendered using our custom software (whose source code is provided at `https://osf.io/amryj`). Using the same text, another choice for edges may have been the semantic relationship between words: this creates a *semantic network*. In Figure 2.2, edges between two words stand for synonymy. In sum, text can be represented by a network in many different ways depending on the goal of the analysis. Goals may include word sense disambiguation [109], or text summarization [106].

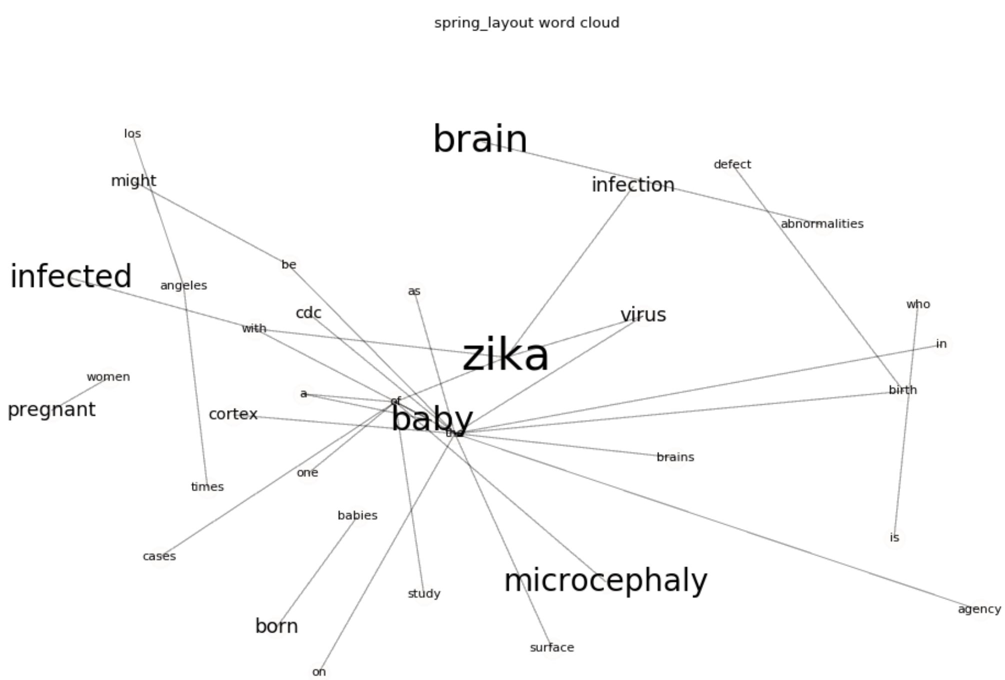


Figure 2.1: A graph showing co-occurrence of words rendered using Spring layout.

Once data has been transformed into a graph, a large number of metrics can be used for analysis [47, 49]. In particular, *centrality* metrics can help the analyst to easily identify structurally important words in the text. Three commonly used such metrics are degree centrality, closeness centrality, and betweenness centrality. These metrics are introduced in definitions 2.2.4, 2.2.5, and 2.2.6 respectively. To illustrate these notions, we computed degree and betweenness centrality on each nodes using the same text as previously, and we rendered the result using a radial layout in Figures 2.2 and 2.3 respectively.

Definition 2.2.4. ***Degree Centrality*** *is the simplest form of centrality. The degree of a node in an undirected graph is simply how many connections it shares with other nodes. The degree centrality of a node is given by: [119, 44]*

$$C_D(v) = \frac{degree(v)}{(n-1)}$$

Definition 2.2.5. ***Closeness Centrality*** *is defined by the shortest path (number of edges) between two nodes. The closeness centrality is the sum of all edges between a node and all other nodes in the graph. Closeness is normalized by the sum of all shortest paths possible (one to each node) [119]:*

$$C_C(v) = \frac{n-1}{\sum_{j \in V} d(v,j)}$$

Definition 2.2.6. ***Betweenness Centrality*** *uses shortest paths. Betweenness is calculated by dividing the number of shortest paths that pass through the node (v) by the total number of shortest paths between all nodes [119, 44]:*

$$C_B(v) = \sum_{s,t \in V} \frac{\sigma(s,t|v)}{\sigma(s,t)}$$

where $\sigma(s,t)$ is the number of shortest (s, t)-paths, and $\sigma(s,t|v)$ is the number of those paths passing through some node v other than s,t. If $s = t$, $\sigma(s,t) = 1$, and if $v \in s,t$, $\sigma(s,t|v) = 0$

Centrality metrics may involve the whole network (in the case of betweenness and closeness) or not (for degree), but they always compute values for specific words/nodes. In contrast, some metrics may provide a number about the network as a whole. Two such metrics are density and average path length, presented in definitions 2.2.7 and 2.2.8 respectively.

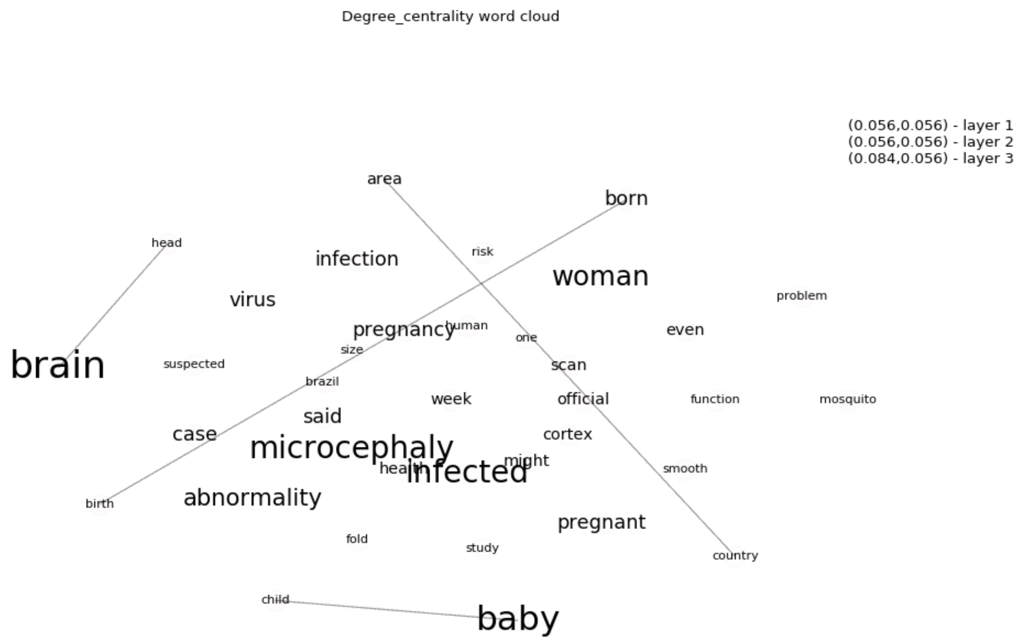


Figure 2.2: A Graph rendered using 3-level Radial layout showing the edges between synonyms .

Definition 2.2.7. *The* ***density*** *of a graph G=(V,E) is the percentage of the number of edges of a clique, observable graph [17, 119, 44]:*

$$D(G) = \frac{2|E|}{|V|(|V|-1)}$$

Definition 2.2.8. *The* ***Average Path Length*** *(also called average distance) is the arithmetic mean of all distances in the graph [17, 46]:*

$$l(G) = \frac{1}{n \cdot (n-1)} \cdot \sum_{i \neq j} d(v_i, v_j)$$

where $d(v_i, d_j)$ *is the length of the shortest path between the nodes.*

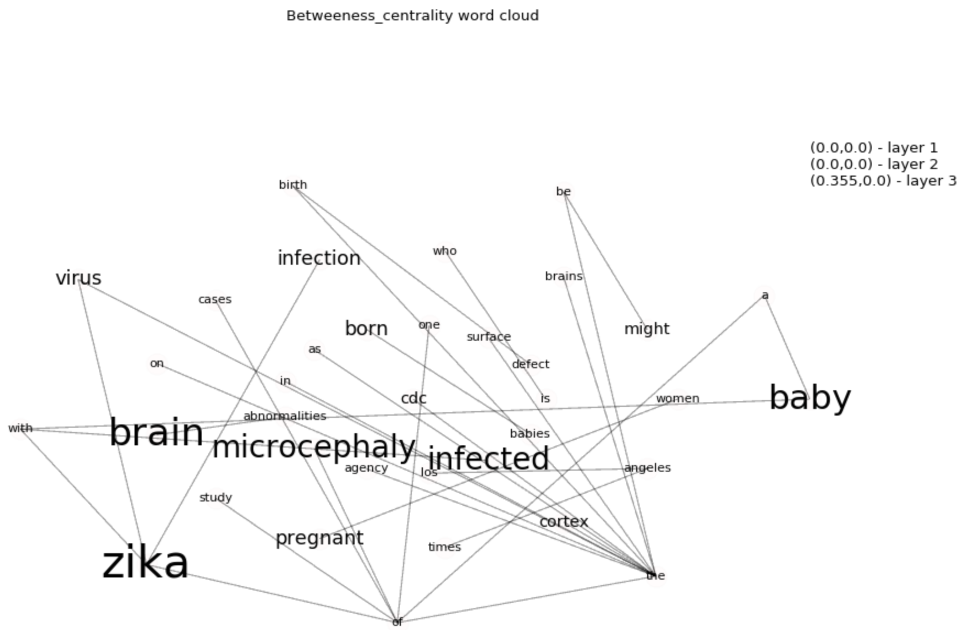


Figure 2.3: A Graph rendered using 3-level Radial layout with words having high Betweenness centrality placed towards the center.

2.3 Fully automatic techniques: word/tag clouds

Word clouds are one of the most commonly used visualization techniques for text. Word clouds are an easy to use yet powerful way to visually summarize a text document by extracting the words with highest frequency from a text or corpus. A related type of visualization is the *tag cloud*. Tags are labels associated to text when manually categorizing them, rather than being automatically inferred from the text itself. Since both types of clouds are about displaying representative terms (either present in the text or user-annotated), they are addressed together in this section.

Over the past few years, most of the research work has gone into the tag cloud visualization. Cui and colleagues have proposed classifying these visualizations into two groups [27]: static tag clouds, and dynamic tag clouds. In their research, the ‘dynamic’ version consists of a static cloud associated to a time slice (Figure 2.4). Users can browse a time series of documents and, for a specific period, a static cloud is generated. Note that this definition of ‘dynamicity’ differs from other visualizations where some elements come and go and those that remain are moved accordingly.

A key problem in tag cloud visualization is that of large white spaces: this is particularly salient in settings where space is scarce (e.g., mobile, PDAs) or shared across many applications (e.g., large displays with numerous renderings shown together). One approach to create a layout for the tags is to think of it as a graph layout (Definition 2.2.3) problem. The problem of creating a layout consists of finding an algorithm that places the nodes to facilitate exploration of a graph, for example by making some properties salient or making it more intuitive for the type of data that it represents. A tag cloud can be transformed into a graph, where tags are nodes and edges stand for the similarity between tags (e.g., measured by co-occurence). Then, a graph layout can be used (e.g., force layout, radial layout) and *only the nodes* (i.e. tags) will be displayed while

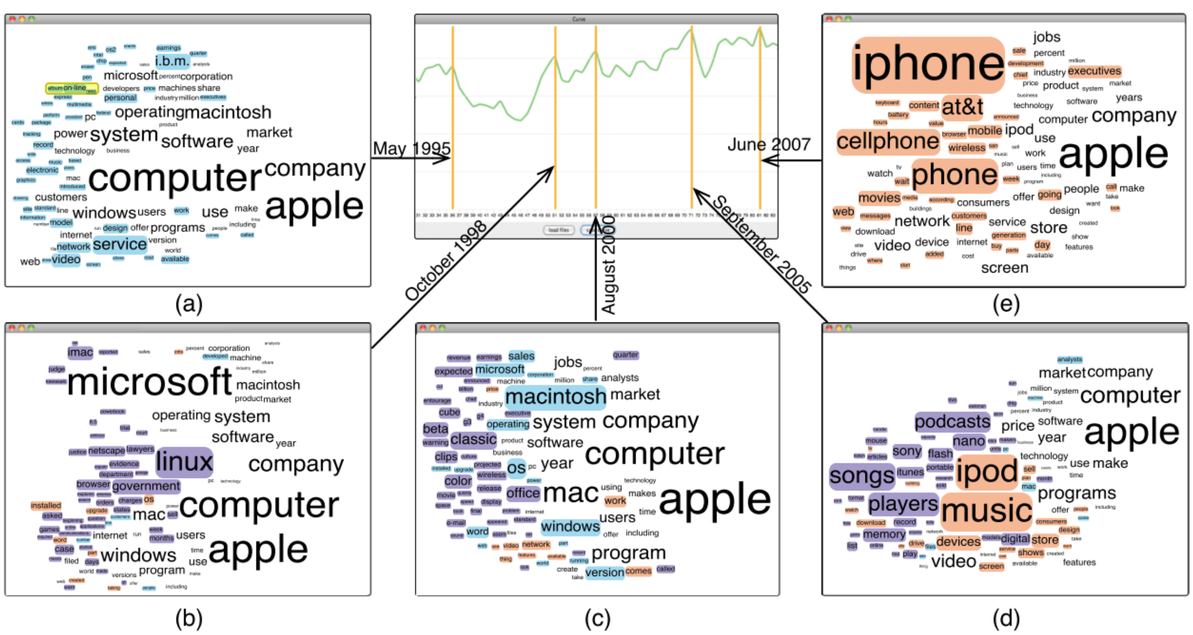


Figure 2.4: Overview of the system, reproduced from Cui et al. [27]. In the top center, the x-axis represents the time (allowing to select a word cloud for a specific time slice) and the y-axis conveys how much of that word cloud is shared with all other word clouds (i.e. ‘significance’).

edges will be ignored. An example is offered in the work of Bielenberg et al. [12], who proposed using the ‘importance’ (defined as the number of occurrences) of words in a radial layout. Given the popularity of tag clouds, there are also several proprietary algorithms to render them, such as those developed for Wordle.net.

Word clouds can serve as a starting point for a deeper analysis [43]. For example they help to quickly judge whether the given text is relevant to a certain topic. In a large corpus of document, a word cloud can be conveniently generated when hovering over one document to gain a succinct sense of its content.

2.4 Semi-automatic techniques: Trees and Summarization

2.4.1 Word tree cloud

The techniques mentioned in the previous section were able to take in text data, and render it without any other human intervention. Tree clouds (Figure 2.5) are similar to word clouds, as they show the frequency of words in a text using the word’s size. There is also a network, which in this case is limited to a tree. The tree’s structure shows the semantic proximity of words in the text. That is, the distance between two words is represented by the length of the path between them in the tree. The other difference between tree clouds and word clouds is that a tree has to have a root, and this is determined by the user. Consequently, we categorized tree clouds as semi-automatic.

According to Gambette and Veronis [43], the first step in building a tree cloud is extracting the list of frequent words from the text. Then, stop words may be removed from the list to get a meaningful tree cloud. In order to connect the words, it is necessary to have distance metrics between pairs of words. This is done using the *semantic distance matrix* (definition 2.4.1), which operates over a sliding window of width w (i.e. number of words; set to 30 by default) and with step s (i.e. distance between two sliding windows; set to 1 by default). Note that words are not stemmed in this procedure, hence 'mosquito' and 'mosquitoes' would both appear on the result.

Definition 2.4.1. *A **semantic matrix** O for two words W_i and W_j is a 2×2 matrix which counts the number of times that both/none/only one appears in a portion of the document (e.g., sentence or paragraph). For example, consider the following text: "Sriram is a computer science student. Northern Illinois University offers a masters program in computer science. Sriram, let us play with the sliding window." Assume a sliding window of width $w = 2$ words and step $s = 1$. In each bigram, we will count whether $W_i = `Sriram'$ and/or $W_j =' science'$ appear. The matrix is:*

	Has 'science'	**Does not have 'science'**
Has 'Sriram'	$O_{i,j}^{1,1} = 1$ ("science. Sriram")	$O_{i,j}^{1,2} = 3$ ("Sriram is", "science. Sriram", "Sriram, let")
Does not have 'Sriram'	$O_{i,j}^{2,1} = 3$ ("computer science", "science student", "computer science.")	$O_{i,j}^{2,2} = 17$

The size of words in tree clouds usually reflects their frequency. Colors can also be used to convey information. For example, Gambette and Veronis use colors to represent categories [43]. Additional information can be displayed using brightness. For example, brightness may be used to encode time (thus making it possible to distinguish articles from different dates). Alternatively, brightness can show whether the word appears in one place in the text or in many places (using a dispersion coefficient). While the method by Gambette and Veronis focuses on encodding information in the words, edges are under-utilized. This suggests that future visualizations can represent information in the edges, either numerical (e.g., through edge thickness) or categorical (e.g., through edge color). This could include semantic distance (numerical) or word relationship (categorical).

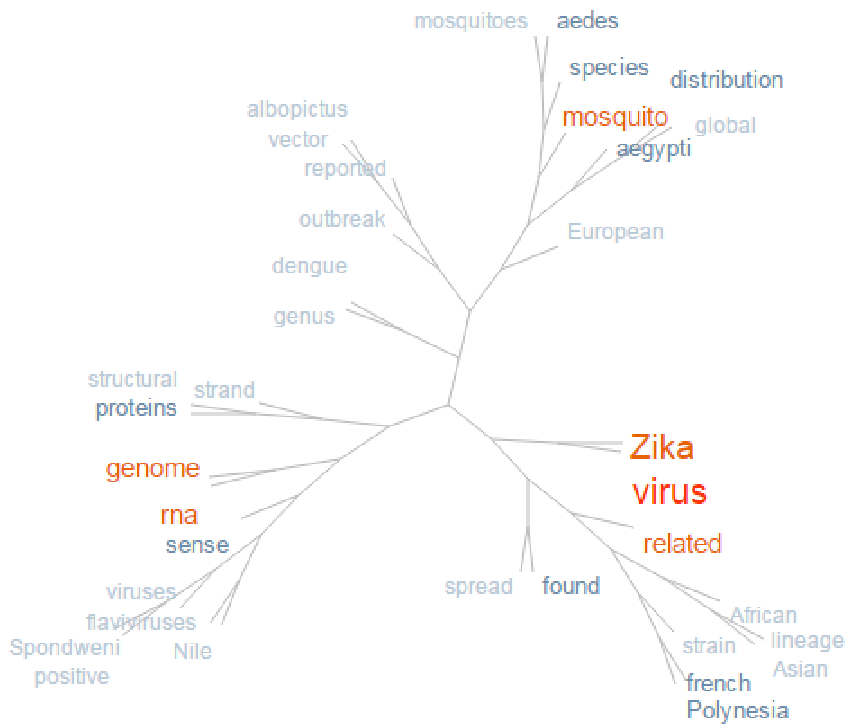


Figure 2.5: Word tree cloud of the Wikipedia ‘Zika’ entry using *http://treecloud.univ-mlv.fr*.

2.4.2 Prefix word trees and suffix word trees

Word trees are a text visualization technique which can be defined as an interactive graphical version of the traditional “Keyword in Context (KWIC)” method [38] proposed by Wattenberg and Viegas [155]. Parts of sentences that follow or precede a set of words chosen by the user are visually organized as a tree. In a *prefix* word tree, the root word is on the left and the words immediately following the root word are to its right. In a *suffix* word tree, the root is on the right, and in a *double* word tree, it is in the center. A double word tree combines the prefix word tree and the suffix word trees. Similar to word tree clouds (section 2.4.1), font size is used to represent the number of times a word or phrase appears. Note that the scale is not linearly proportional

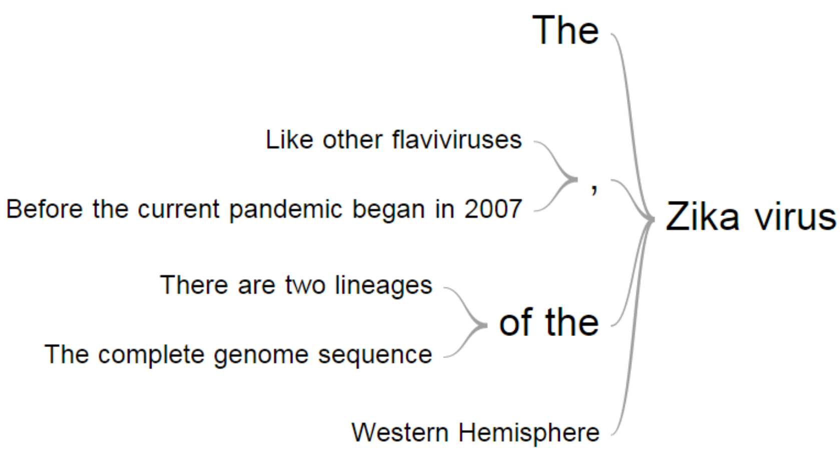


Figure 2.6: Suffix Word Tree showing all instances of “Zika virus”.

to the word’s frequency: instead, the word’s size is proportional to the square root of the word’s frequency [155], so that it leaves sufficient room above and below.

The word trees shown in this section are generated with the help of Google developer charts. The content is the same as in the previous section, regarding the Zika virus. Figure 2.6 shows an example of a suffix word tree, while a prefix word tree is displayed in Figure 2.7 and a double word tree is illustrated in Figure 2.8.

With the use of Word trees, it is easy to spot repetition in contextual words that follow a phrase. A word tree can either narrow or widen the text search. For instance, if the current phrase is “Zika virus”, clicking on the initial “Zika” will re-center the tree on the phrase “Zika” (see Figure 2.6) thus helping the user to explore the context further. On the other hand, if the user clicks on a word in a branch of the tree, such as “2015-present” in the branch “outbreak”, then the tree will be re-centered on the longer phrase, “outbreak (2015-present)” (Figure 2.6).

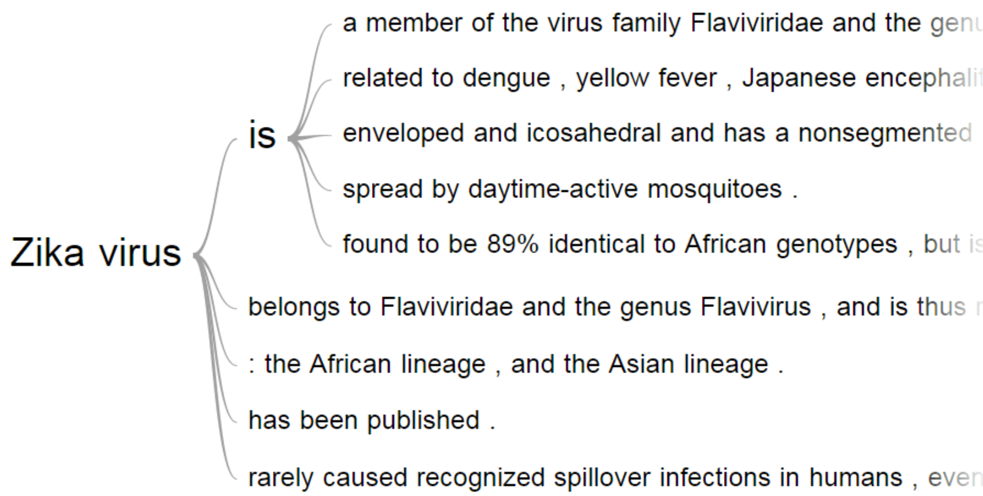


Figure 2.7: Prefix word Tree showing all instances of "Zika virus".

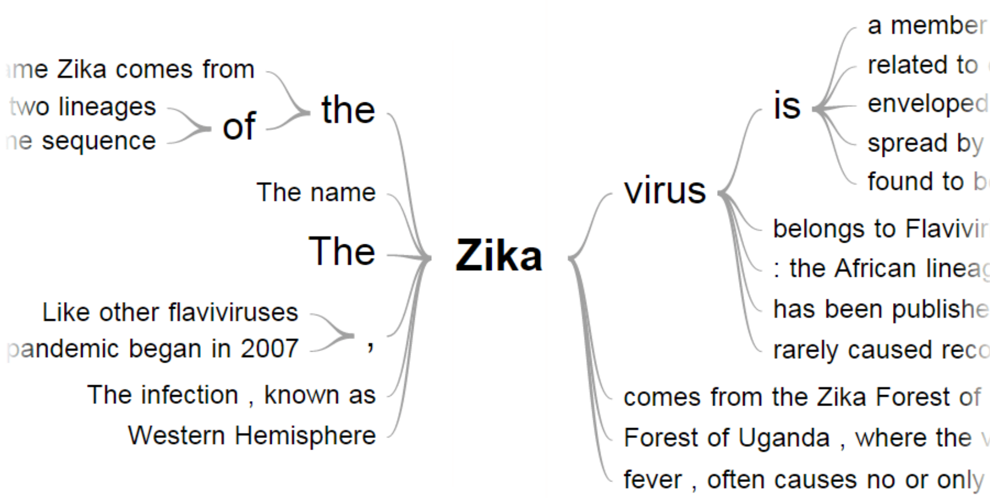


Figure 2.8: Double Word tree rooted at "Zika".

2.4.3 Text summarization

The difficulty in understanding enormous texts has necessitated intensive research in the area of Text summarization within the Natural Language Processing community. Automatic Text summarization is the process of generating a summary by condensing large articles into short paragraphs or sentences while retaining the important information.

Summarization is a hard problem in natural language processing because the summary of a document depends on the context in which the summary is needed. At the same time, it is tedious for human beings to manually summarize large texts and essays. Consequently, the growth in data is paralleled by a growing need for automatic summarization, even when the techniques are imperfect. There is a wide variety of techniques to summarize text depending on the type of the text and the type of summary required. To simplify them, we consider mainly two approaches (while noting the existence of hybrid approaches as well): extraction (also known as *shallow techniques*) and abstraction (also known as *deep techniques*) [120]. The summary generated by extraction is developed based on the words and phrases in the actual text. Some smoothing techniques are used to repair any incoherence occurring in such extractions. The summary generated by abstraction techniques is not limited to the explicit words of the text: natural language generation techniques are used to generate the summary [120]. Since the output texts are machine generated, summarization may require rich linguistic sources such as WordNet (a lexical database which groups words into sets of synonyms called *synsets*) and domain specific corpora. In this thesis, we use extraction-based methods. Consequently, the remainder of this section is devoted to identifying which existing sentences in the documents should be selected to be part of the summary.

Most early work on single-document summarization was focused on technical documents. In his seminal 1958 paper, Luhn proposed a text summarization technique which describes the research done at IBM in 1950s [96]. In particular, he proposed an algorithm which produces a summary based on the frequency of the words, by assuming that frequent words represent the most important concepts of the text. The process is summarized in Algorithm 1 and works as follows. First, we sort the words by frequencies. Then, a significance factor is computed for each sentence by counting how many frequent words it contains. Sentences are then ranked by significance factor. Finally, the top ranked sentences are selected to constitute the summary. Note that this approach only looks at the frequency of the words: it loses the ordering of words within sentences. This representation of words as frequencies without ordering is called a *bag of words*.

Algorithm 1 Luhn's algorithm

```
procedure SENTENCESIGNIFICANCE(S)
    count ← {}                                        ▷ Frequencies of words
    sentenceSignificance ← {}                         ▷ Significance Factors of sentences
    for sentence i ∈ S do
        for word w ∈ i do
            if w ∈ Count then
                count[w] ← count[w] + 1
            else
                count[w] ← 1
    for sentence i ∈ S do
        for word w ∈ i do
            if i ∈ sentenceSignificance then
                sentenceSignificance[i] ← sentenceSignificance[i] + count[w]
            else
                sentenceSignificance[i] ← count[w]
    return sentenceSignificance
```

There are several issues with Luhn's algorithm. First, it does not take into account the position of words in the document. This ignores that paragraphs such as the first and last (concluding) ones may contain more important information than other paragraphs. Second, it is limited to a single document. Third, it ignores semantics. One solution provided by Erkan and Radev that helps to summarize a whole corpus consists of identifying 'centroids', and then considering that sentences that have more words from the centroid are better [34]. Algorithm 2 shows how the centroid is computed. An important statistical notion involved in this algorithm is the term frequency-inverse document frequency (*tf-idf*). While there are different ways to operationalize the term frequency *tf*, we use its simplest (not normalized) form consisting of the raw frequency of a term. The inverse document frequency *idf* is also taken in its simplest form, without smoothing etc. Each is formally presented in definition 2.4.2, together with their combination as *tf-idf*.

Definition 2.4.2. *The **term frequency** $tf(t,d)$ is the number of times that a term t appears in one document d. The **inverse document frequency** $idf(t,D)$ looks at whether the term is common across the set of documents D. If we denote by n_t the number of documents containing the term t, then $idf(t,D) = \log\frac{|D|}{n_t}$. The **tf-idf** shows the importance of the term t in a document d given the corpus D:*

$$tfidf(t,d,D) = tf(t,d) \times idf(t,D)$$

Algorithm 2 Centroid Score Computation

```
procedure CENTROIDSCORECOMPUTATION(Sentences S, threshold t)
    words_tfidf ← ∅
    for s ∈ S do
        for w ∈ s do
            words_tfidf[w] ← tf(w) × idf(w)
    words_centroid ← ∅
    for w ∈ words_tfidf do
        if words_tfidf[w] > t then
            words_centroid[w] ← words_tfidf[w]
        else
            words_centroid[w] ← 0
    sentenceScore ← ∅
    for s ∈ S do
        sentenceScore[s] ← 0
        for w ∈ s do
            sentenceScore[s] ← sentenceScore[s] + words_centroid[w]
    return sentenceScore
```

To go beyond this, sentences can be seen as a graph using the *sentence based graph* representation (Definition 2.4.3) [34]. Sentences similar to many other sentences can intuitively be thought as providing a good 'proxy', or summary. There are different ways to operationalize such sentence based graph. Here, each node represents a sentence as a vector (similarly to a bag-of-word) but instead of the raw frequencies as in Luhn's approach, each word is given its tf-idf. The similarity between two sentences (i.e. the edge's value) is then computed using the cosine similarity, which can be seen as a method of normalizing the document length during comparison.

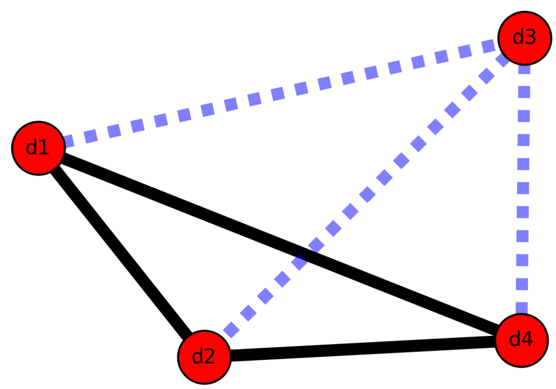


Figure 2.9: A graph rendered with documents as nodes and the similarity between them as edges. Similarity is computed using cosine similarity. Dotted lines denote edges with similarity scores less than 0.1.

For example, consider 4 documents $d1$ = "Tom reads novels", $d2$ = "Tom and Harry read newspapers", $d3$ = "The sun in the sky is bright", and $d4$ = "Tom and Harry enjoy reading newspapers". Figure 2.9 renders a network of these documents with the edges representing the similarity between them.

Definition 2.4.3. *In a **sentence based graph**, the nodes represent sentences. An edge connecting two nodes represents the similarity between the two sentences.*

We note that, while these methods can deal with summarizing a corpus rather than a single document, they still have many drawbacks in sentence selection. For example, they still do not weight sentences based on their positions in documents. In addition, redundancy is not taken care of. For example, imagine that the same sentence is present in two documents (e.g. as a quote from a spokesperson). The algorithm will see this case as *two* sentences with high similarity, which are thus more likely to be present in the summary. Finally, there is still no semantic, so polysemous words (i.e. words which have different meanings) are treated as the same words. For example, two distinct meanings of the word "cold" are (1) lacking warmth of feeling and (2) a common viral infection in which the mucous membrane of the nose and throat become inflamed.

Semantics, and particularly word sense disambiguation, are an important problem in the natural language processing community. One strategy was presented by [7], using the notion of *lexical chain*. A lexical chain is a set of words (from the text) that are inter-related in one of three ways:

(1) Extra strong relation: between a word and its literal repetition. For example, consider the following text "**Arm** is a kind of limb. **Arm** workouts improves muscle strength.".

(2) Strong relation: between two words connected by a word-net relation. In the example above, the word *Arm* is a hyponym to *limb*. Hyponymy is a wordnet relation where a more specific word is connected to a given word.

(3) Medium-strong relation: between two words when the link between synsets of words is longer than one with not more than five links [74]. Consider the sentence "I have a dog and a cat." Here Dog and cat are hyponyms to 'animal' but they are not directly related to each other. Hence there is a medium-strong-relation between them.

All three types are illustrated in Figure 2.10. Algorithm 3 provides the method to construct a lexical chain.

Identifying the right chain for a word is also based on the distance between the word and the words in the chain. The authors suggest that the maximum distance between related words should depend on the type of relation: no limit for extra strong relations, a window of 7 sentences for strong relations, and a window of 3 sentences for medium strong relations. Note that 'a window' is always measured as going forward, i.e. if we start a one sentence then a "window of 3 sentences" includes this sentence and the next two.

When a given candidate word has a word net relations with words from different chains, It is added to a chain based on the following priority order: Extra-strong relations are given the

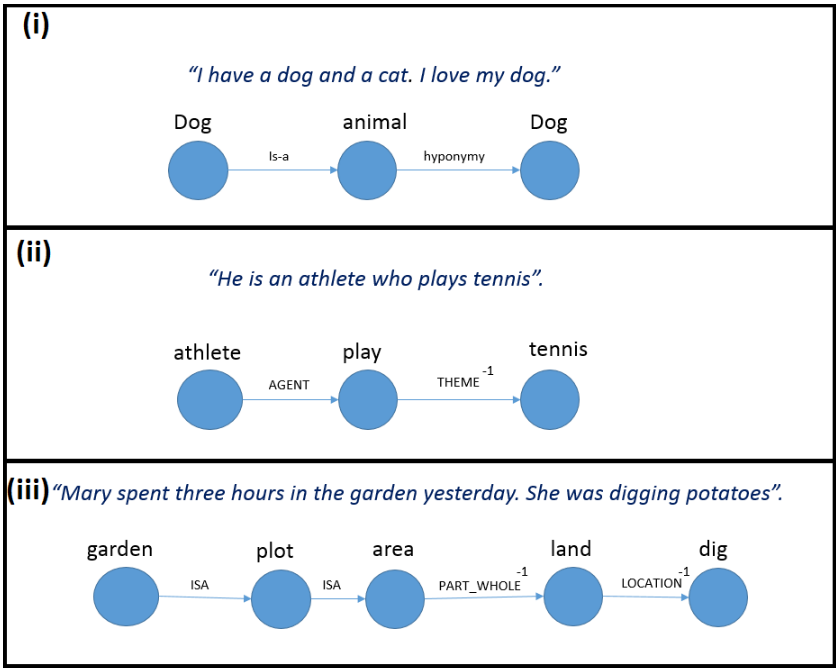


Figure 2.10: Lexical Chains showing different relationships. Note that '-1' denotes a reverse order, e.g. a land is an area. (i) shows an extra strong relation between 'dog' and its repetition. (ii) represents a strong relation between 'play' and 'athlete', and between 'play' and 'tennis'. (iii) shows a medium strong relation between 'dig' and 'garden'.

highest priority and Strong relations are preferred over medium-strong relations. Intuitively, if a word w_1 has a medium strong relation with another w_2 which is 2 sentences after w_1, and a strong relation with w_3 which is 6 sentences after w_1. w_1 is assigned to the chain having w_3 and not to the chain with w_2. A thorough example of the construction is provided in Figure 2.11 based on the following text, taken from `http://betobaccofree.hhs.gov/laws/`: "**Tobacco** control programs aim to reduce **disease**, **disability**, and **death** related to **tobacco** use. A comprehensive approach—one that includes educational, clinical, regulatory, economic, and social strategies—has been established as the best way to eliminate the negative **health** of **tobacco** use."

This process can assign a given word to several chains. To reduce the algorithmic complexity, when the number of chains for a particular word exceeds a given threshold (i.e. 10 chains), then pruning is performed by removing interpretations with low scores from the chaining process. De-

Algorithm 3 Algorithm for Chain Construction

```
procedure ChainConstruction(S)          ▷ Reduces sentences to only containing nouns
    CandidateWords ← ∅
    for s ∈ S do
        for w ∈ s do
            if POS(w) is Noun then          ▷ If Part-of-speech tagging identifies w as a noun
                CandidateWords ← CandidateWords ∪ {w}
                                    ▷ Assume that CandidateWords can be indexed from 0 to n
    chains ← ∅
    for i ∈ [0, n] do
        chain ← {CandidateWords_i}
        for j ∈ [i + 1, n] do                              ▷ Applies a sliding window
            if CandidateWords_j ∈ wordnet.synset(CandidateWords_i) then
                chain ← chain ∪ {CandidateWords_j}
        chains ← chains ∪ chain
    return chains
```

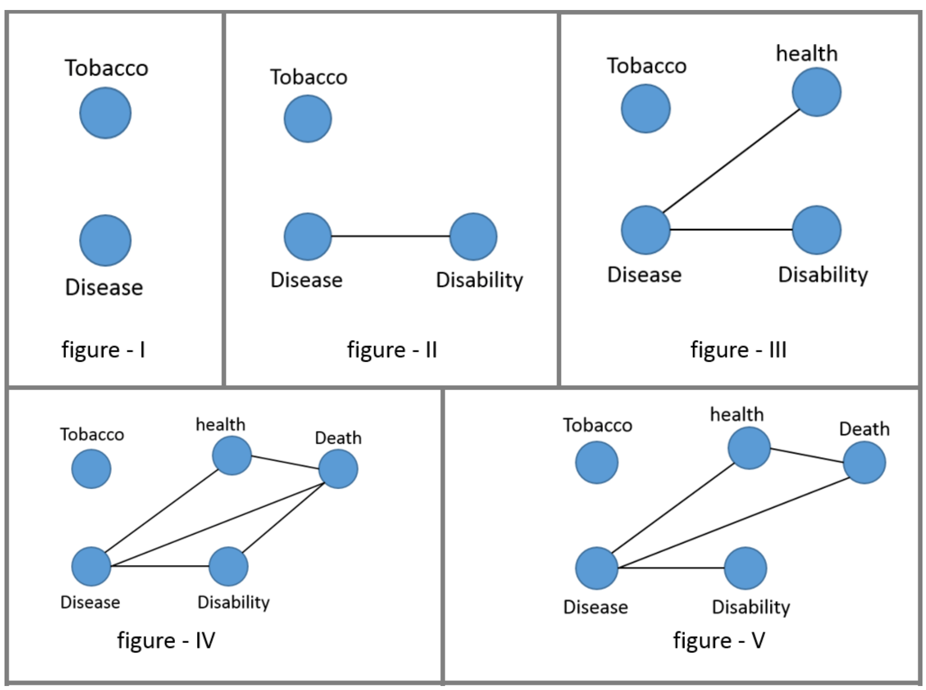


Figure 2.11: Visual Representation of lexical chains.

tails of the equations can be found in [7]. Once the chains have been constructed, we can extract sentences from them and start forming the summary. To do so, Barzilay suggests (i) finding a

Table 2.1: Comparison of text summarization techniques.

Summarization Technique	Pros	Cons
Luhns Word Frequency based summarization [96]	Frequency of occurrence of words is an important metric to generate the summary.	Different parts of the text are given the same weightage and Words which are not related to the context are also included.
Erkan and Radevs Graph based summarization [34]	Sentence similarity is computed so that sentences which are not related to the context of the documents are eliminated.	Words with different senses are treated as the same.
Barzilay and Elhadads Lexical chains based text summarization [7]	Barzilay was the first to discuss Word sense disambiguation which is one of the most important problems in NLP.	Extracted sentences contain anaphora links (words that are used instead of repetition) to the rest of the text. For example: *"His Father had cancer. He might get it as well."* This has been investigated in [112]. Sentence granularity: all our methods extract whole sentences as single units. This has several drawbacks: long sentences have significantly higher likelihood to be selected.

representative word from each chain (i.e. the most frequent term), and (ii) extracting the sentence from the text that first contains this word. This results in extracting one sentence per chain.

2.5 Supervised Techniques: Classification and Document Retrieval

2.5.1 A brief introduction to Classification

In the field of machine learning, a classification model is constructed using a set of training observations $D = \{X_1, X_2, \cdots, X_N\}$ such that each observation has a class value (or 'label') taken from a discrete set of size m. When $m = 2$, we speak of *binary* classification. Conversely, when

$m > 2$ we are in the general *multi-class* situation. The classification model relates the features in the training data to one of the class labels. The model can then be used on new instances (i.e. for which the label is unknown) in order to predict the label. One usually speaks of a 'classification' problem when labels have categorical values, and of a 'regression modeling' problem when labels have continuous values. Classification can be applied to generate insights from datasets of various sizes and data types, ranging from large-scale national (structured) surveys [48] to pilot-studies with questionnaires [26, 25] and text data. Our focus is on text classification, which is a well-known problem whose examples abound (e.g., news filtering, opinion mining, spam identification) [1].

Before any classification task, one of the most fundamental tasks that needs to be accomplished is that of document representation and feature selection (i.e. identifying the features relevant for the classification task). While feature selection is also desirable in other classification tasks, it is especially important in text classification due to the high dimensionality of text features and the existence of irrelevant (noisy) features. Different measures to identify the importance of features were proposed in the literature [1]. These include Gini index or entropy, information gain, mutual information, χ^2- statistic, etc.

The characteristics of an object in the data set are known as the *feature values* and are typically presented to the machine in a vector called a *feature vector*. A classifier is a type of supervised learning, because it is built on *training data*, that is, instances that are labelled with the target class [13]. When the classifier is built, the vectors are considered in isolation from all others [13]. Intuitively, the classifier then seeks combinations that will help to correctly identify the labels based on the input data. In order to evaluate the quality of a classifier, there are three basic metrics: precision (Definition 2.5.1), recall (Definition 2.5.2), and the F1 score (Definition 2.5.3).

Definition 2.5.1. ***Precision*** *is the ratio* $\dfrac{tp}{(tp + fp)}$ *where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative [102].*

Definition 2.5.2. ***Recall*** *is the ratio* $\frac{tp}{(tp + fn)}$ *where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples [102].*

Definition 2.5.3. *The* ***F1 score*** *can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal [60].*

$$F1 = \frac{2 * (precision * recall)}{(precision + recall)}$$

There are many ways to build a classifier, and several of them may end up having the same ‘quality’ when judged by the metrics aforementioned. For example, a linear classifier makes a classification decision based on a linear combination of the inputs. Alternatively, a Support Vector Machine (SVM) builds a hyperplane. As shown in figure 2.12, there are different lines (or hyperplanes) that offer a solution to the classification problem. A line which passes too close to points will capture noise (over-fitting) and hence it is not an optimal solution. The goal of an SVM algorithm is to find the hyperplane that has the largest minimal distance to the training data points. The notion of *margin* in SVMs refers to double that distance. The optimal solution for this example is shown in figure 2.13. In this thesis, we will frequently employ SVMs.

While some applications of text classification have received particular attention, such as *sentiment analysis*, one should note that they really are typical text classification problems. In sentiment analysis, a classification model is constructed using a set of training observations, as in any other classifiers. The particularity is that the class labels refer to ‘sentiments’.

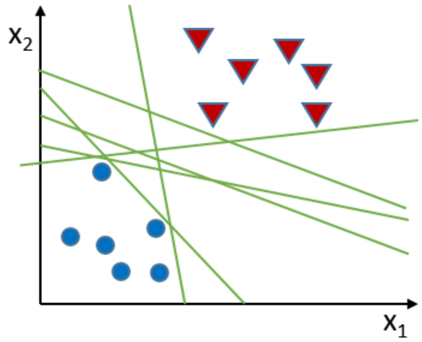


Figure 2.12: Multiple solutions to the problem.

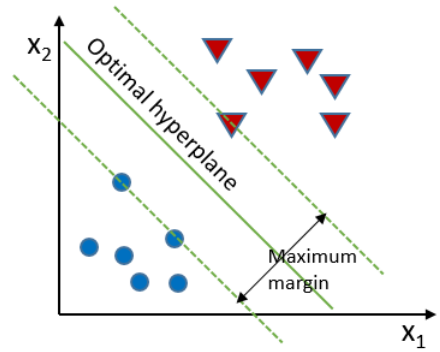


Figure 2.13: Optimal Hyperplane.

2.5.2 Document retrieval

The goal of an information retrieval system is to return documents optimally based on a query. The system achieves this goal by returning documents which are relevant. In other words, documents having higher scores than others are deemed to be relevant and thus retrieved. The retrieval accuracy depends on the efficiency of the scoring function. Many retrieval models have been proposed over decades, but no single retrieval function has been optimal [161]. Among these numerous models, typical ones include variants of the vector space model [135, 39, 136] and probabilistic models [42, 94, 115, 125]. The remainder of this section succinctly introduces both types in turn. Then, we will explain language modeling approaches such as the query likelihood model and the KL divergence model. We will also discuss the advantage of using topic modeling over unigram models.

The Vector space model was developed for the SMART information retrieval system [134] by Gerard Salton and his colleagues [137].*Vector space model* treats documents and queries as vectors

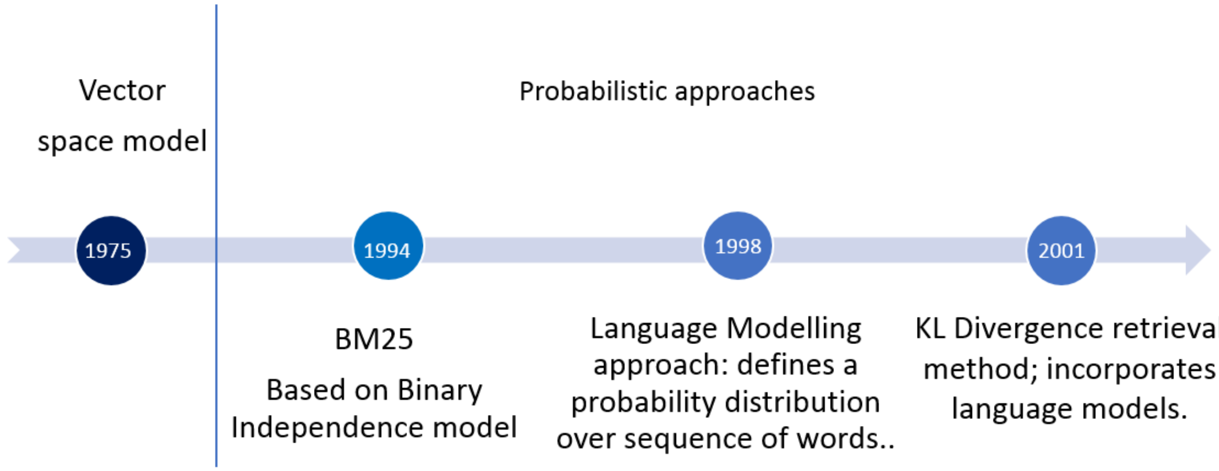


Figure 2.14: Document Retrieval time line showing Vector space models [137], BM25 [125], Language modeling approach [115], and KL Divergence retrieval technique [93].

in an N-dimensional space, where N is the number of indexing features. That is, the j-th document of a collection is represented as a document vector:

$$d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$

where the weights are non-zero if the corresponding term appeared at least once in the document. One way of computing the weights is to use the tf-idf (see subsection 2.4.3 for details). Similarly, the user's query is given by the vector

$$q = (w_{1,q}, w_{2,q}, \ldots, w_{n,q})$$

Generally, the document vectors are ranked according to their cosine similarity (or some other distance function) with the query vector [140, 10], computed as:

$$cos(d, q) = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2}\sqrt{\sum_{i=1}^{N} w_{i,q}^2}}$$

The vector-space model with tf-idf weighting and document length normalization has traditionally been one of the most effective retrieval models, and it remains quite competitive as a state of the art retrieval model [161].

The popular BM25 (usually called *Okapi*) retrieval function [125] is very similar to the tf-idf vector space retrieval function discussed above, but it is motivated and derived from the 2-Poisson probabilistic retrieval model [125, 126] with heuristic approximations. 'BM' stands for "Best Match" and the 25 is the version number showing the evolution of this weighting scheme. Consider a document $D = (df_1, \dots, df_V)$ with a vocabulary of size V (i.e. number of distinct words), where df_j is the frequency of the jth term in the document. To score the document D against a query, BM25, a term weighting function $w_j(D, C)$, exploits term frequency, document's length, and collection statistics. For ad-hoc retrieval, by ignoring repetition of terms in the query, this function can be expressed as:

$$w_j(D, C) = \frac{(k_1 + 1)df_j}{k_1((1-b) + b\frac{|D|}{|\bar{D}|}) + df_j} \log \frac{N - df_j + 0.5}{df_j + 0.5}$$

where d_j is the frequency of term j in the document, $|D|$ is the length of the document (i.e., the total count of words in D), $|\bar{D}|$ is the average length of the document in the collection C, k_1 and b are free parameters. The document score is obtained by adding the term weights of the words that match the query q:

$$W(D, q, C) = \sum_{j \in q \cap D} w_j(D, C).q_j$$

While both models rely on heuristic design of retrieval functions, an interesting category of probabilistic models called *language modeling approaches* have been efficiently retrieving documents without much heuristic design, that is, they employ less or no free parameters compared to their vector space counterparts. A language model refers to a probabilistic model of text: it defines a probability distribution over sequences of words. Ponte and Croft proposed to score documents using language modeling technique [115], which later became known as the *query likelihood scoring method*. In their approach, a language model is estimated for each document, and then the documents are ranked based on their likelihood of the query according to their estimated language models. Ponte and Craft used a basic language modeling approach in which the query is assumed to be a sample of words drawn according to a document's language model. Documents are ranked based on the probability score given to the query. Intuitively, a query is given highest score if its words occur frequently in the document. Retrieval based on language models can explain the data by means of probabilistic modeling [115], which in turn can help in improving the retrieval performance. Vector space models cannot provide such guidance to the researcher.

The query likelihood retrieval function can be formally described as follow. Let *D* be a document and *q* be a query. Let θ_D be a language model based on document *D*. The score of the document *D* with respect to *q* is the conditional probability $p(q\|\theta_D)$. Let $V = \{w_1, ..., w_{|V|}\}$ be the vocabulary of the collection C. We can define a binary random variable $X_i \in \{0, 1\}$ for each word w_i to indicate whether the word w_i is present $(X_i = 1)$ or absent $(X_i = 0)$ in the query. According to this model, the query likelihood can be written as

$$p(q\|\theta_D) = \prod_{w_i \in q} p(X_i = 1\|D) \prod_{w_i \notin q} (1 - p(X_i = 1\|D))$$

The maximum likelihood (ML) estimator, $p(X_i = 1\|D)$ is equal to the relative frequency of the word w_i in D:

$$p(X_i = 1\|D) = \frac{c(w_i, D)}{|D|}$$

where $c(w_i, D)$ is the frequency of word w_i in document D and $|D|$ is the length of D.

One problem with the ML estimator is that an unseen word in a document D would get a zero probability and the model accuracy is negatively affected. To avoid this, the ML estimator needs smoothing. In Ponte and Croft's model, smoothing of setting the probability of a missing word *in a* *document* as the probability of the missing word *over the whole collection*. Consequently, none of the words in the collection would get a zero probability. They further made the smoothing robust, by heuristically taking the geometric mean of the ML estimate and the average term frequency across all the documents in the collection [115].

The basic language modeling approach (i.e., the query likelihood scoring method) can be instantiated in different ways by varying (1) θ_D (e.g., multiple Bernoulli or multinomial), (2) estimation methods of θ_D (e.g., different smoothing methods), or (3) the document prior $p(D)$.

The state-of-the-art probabilistic model is the *KL-divergence retrieval model*, which is a robust and empirically effective document retrieval model that can incorporate advanced language models [93]. The KL divergence [91] is a statistical measure that quantifies how close a probability distribution is to another probability distribution. The KL-divergence retrieval method was introduced in [93]. It is similar to vector space model but we use language models rather than using term vectors to represent documents and the query [162]. This approach considers two language models: one for the query (θ_Q), and one for the document (θ_D). In other words, a specific query is a sample observed from a query language model (θ_Q), while a specific document is a sample from a document language model (θ_D). Intuitively, the smaller the distance between the document model and the query model, the higher the document would be ranked.

The KL-divergence of these two models is measured to see how close they are to each other, and that distance is used as a score to rank the documents. Formally, the score of a document D for a query Q is given by:

$$s(D, Q) = \sum_{w \in V} p(w \| \theta_Q) \log p(w \| \theta_D) - \sum_{w \in V} p(w \| \theta_Q) \log p(w \| \theta_Q)$$

Smoothing plays two roles in this retrieval function [160]:

(1) estimation improvement: it helps improve our estimate of θ_D when D is small, and

(2) query term discrimination (IDF): it helps reducing the weights of common terms in the query.

IDF weighting in the query likelihood retrieval model is achieved indirectly through smoothing with the collection language model $p(w\|C)$ [160]. Dirichlet prior smoothing has been recognized as an effective smoothing method for retrieval [163]. The KL-divergence retrieval model together with Dirichlet prior smoothing is the state-of-the-art baseline method for the language modeling approaches to information retrieval. The probability of term w in document D after dirichlet prior smoothing is:

$$P(w\|D) = (1-\alpha)P(w\|D) + \alpha P(w\|C)$$

where $\alpha = \frac{\mu}{|D|+\mu}$, and μ is set to maximize a retrieval metric for a set of queries and a collection of documents.

CHAPTER 3

VISUALIZING TEXT DOCUMENTS

In the previous chapter, we have seen different text visualization and Text summarization techniques to understand and analyze text. As a result of the vast increase in the amount of unstructured data, there is significant increase in the need for text analytics. Staggering volumes of textual information is one problem policy makers face when trying to understand the outcome of a new intervention. In this chapter, we will describe how text visualization techniques and automatic text summarization can be helpful in framing health interventions. We exemplify the usage with the help of a case study centered on taxes on Sugar Sweetened Beverages (SSB) in California. Our Scripts and full sized images can be accessed online at *https://osf.io/3x6av/*.

All of this chapter was published in the following article [51]:

- Philippe J Giabbanelli, Jean Adams, **Venkata Sai Pillutla**. Feasibility and Framing of Interventions Based on Public Support: Leveraging Text Analytics for Policymakers. In *Social Computing and Social Media*, Lecture Notes in Computer Science 9742:188-200, 2016.

My contributions consisted of (i) producing the visualizations, (ii) implementing and applying text summarization techniques. Data was collected by PJG, and expertise on the context was provided by JA.

3.1 Introduction

Public opinions play an important role in planning and implementing new policies. Understanding the feasibility and acceptability of a public policy is important for the policy makers when they frame an intervention. Understanding public opinions in policy making can not only help policy makers frame interventions in a better way but also implement them in a publicly acceptable way.

Close to two thirds of US adults are currently overweight or obese [87]. Several policies have been proposed to tackle the obesity epidemic [164, 54, 150]. These include economic measures such as increasing taxes on unhealthy food items and providing subsidies for healthier ones [6]. There is growing evidence from modelling and pragmatic studies [35, 18] that taxes on less healthy food can reduce in-take and sales. But, many of these population interventions cannot gain public acceptability, meaning policy makers choose not to implement them [117]. Consequently, there is a need to better understand public opinions regarding these sorts of public health interventions. Surveys and qualitative analysis have been used to understand public opinions [124]. The drawback with surveys and qualitative analysis is they need time to be deployed and analyzed.

Twitter has been a popular choice for many studies as Twitter messages ('tweets') [133, 15, 79, 32]. But the tweets are limited to 140 characters which limits the amount of information people convey in the health policy debate. While news articles can be a better alternative as they convey more meaningful information along with the user comments.

3.2 Case study

Despite modelling studies showing evidence that SSB taxes can have beneficial health effects [35, 18], concerns over public acceptability of such policies are one reason why policy makers are reluctant to consider framing such policies. The two SSB taxes in our case study are put to the vote in 2014 in Berkeley and San Francisco, California. An earlier 2011 survey in nearby Santa Clara county found that 67% would support a SSB tax and 37% would oppose it [143]. In Berkeley, CA, the tax was $0.01 per ounce on the distributors of Sugar Sweetened Beverages (SSB), and syrups operating within the city. The proposal was for a tax that would apply to sugary soda, energy drinks, juice with added sugar, and syrups that go into sugary drinks. 100% juice and drinks with milk as the first (primary) ingredient were exempt because of their nutritional value. Diet soda and alcoholic drinks were also exempt. The tax revenue was designed to go into the city's general fund, and a Sugar-Sweetened Beverage Product Panel of Experts had to publish an annual report forming recommendations on how to allocate the funds to "reduce the consumption of Sugar Sweetened Beverages in Berkeley and to address the results of such consumption" [147]. On 11 February and 4 February, the decisions to vote on SSB tax through a formal ballot were made in Berkeley and San Francisco respectively. There was campaigning from both supporters and criticizers of SSB tax. Ultimately, the tax was put to vote and gained support from 76.16% of voters in Berkeley while it did not pass in San Francisco.

3.3 Creating a Corpus: data collection and cleaning

3.3.1 Data collection

The time period of interest and the main events are summarized in Figure 3.1. We collected news reports published between 1 January 2014 and 31 January 2015. We captured any coverage related to the run-up to the ballot decision by extending the data collection period back to 1 January 2014. Extending the inclusion period to 31 January 2015 gave the opportunity to capture short term reflections of the implementation.

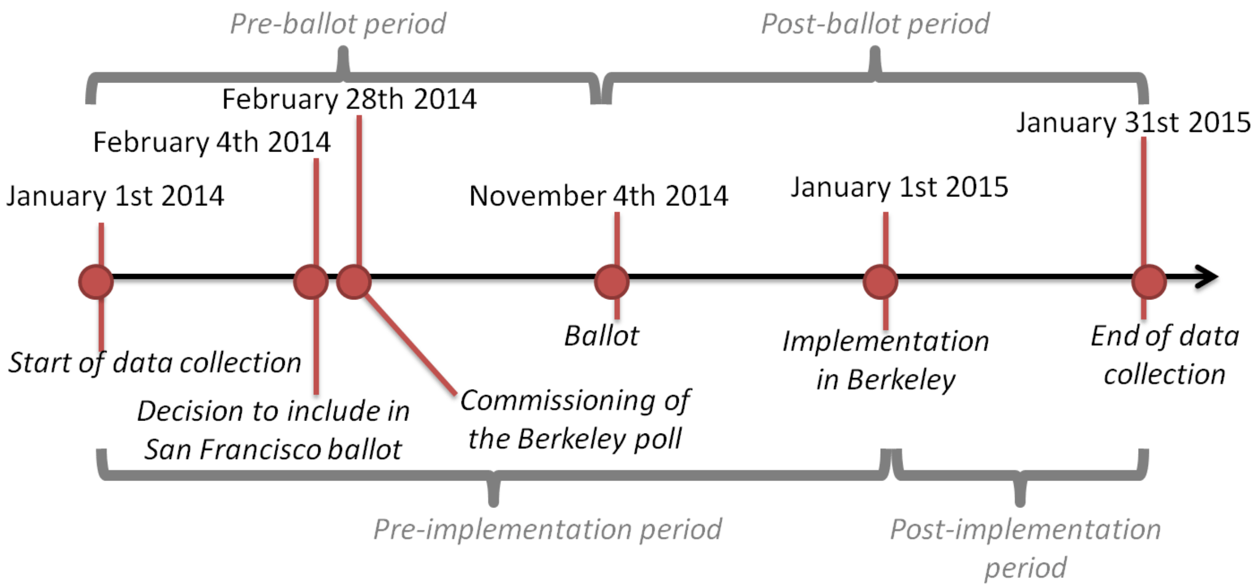


Figure 3.1: Timescale for the data collection and main periods used in the analysis.

In order to explore any changes in reporting over the time-periods before the ballot, after the ballot but before implementation, and after implementation, reports were considered in three groups. Reports published between 1 January 2014 and 4 November 2014 were *pre-ballot*; reports published between 5 November 2014 and 31 December 2014 were *post-ballot* but *pre-implementation*; and reports published between 1 January 2014 and 31 January 2015 were *post-implementation*. Creating these categories allows to explore the impact of events during analysis.

We performed text-analysis on all types of news articles (including news, features, editorials and other comment) as well as reader comments. Reader comments provide an insight into public perception and can show the divide between the arguments in the article and the public opinion. Newspapers were selected if they published at least 4 articles between 1 January 2014 and 31 January 2015 that matched our target content, per rules summarized in Box 1. Candidate newspapers included local and national American newspapers, as well as international English-language newspapers. Four successive approaches were used in identifying candidate newspapers, resulting in a total of 9 newspapers with 165 news articles and 3,864 comments (Figure 3.2). (Figure 3.2 shows the time at which these articles were written, showing that articles appear around the elections as witnessed in previous policy research [110].

Box 1. Selection Criteria.

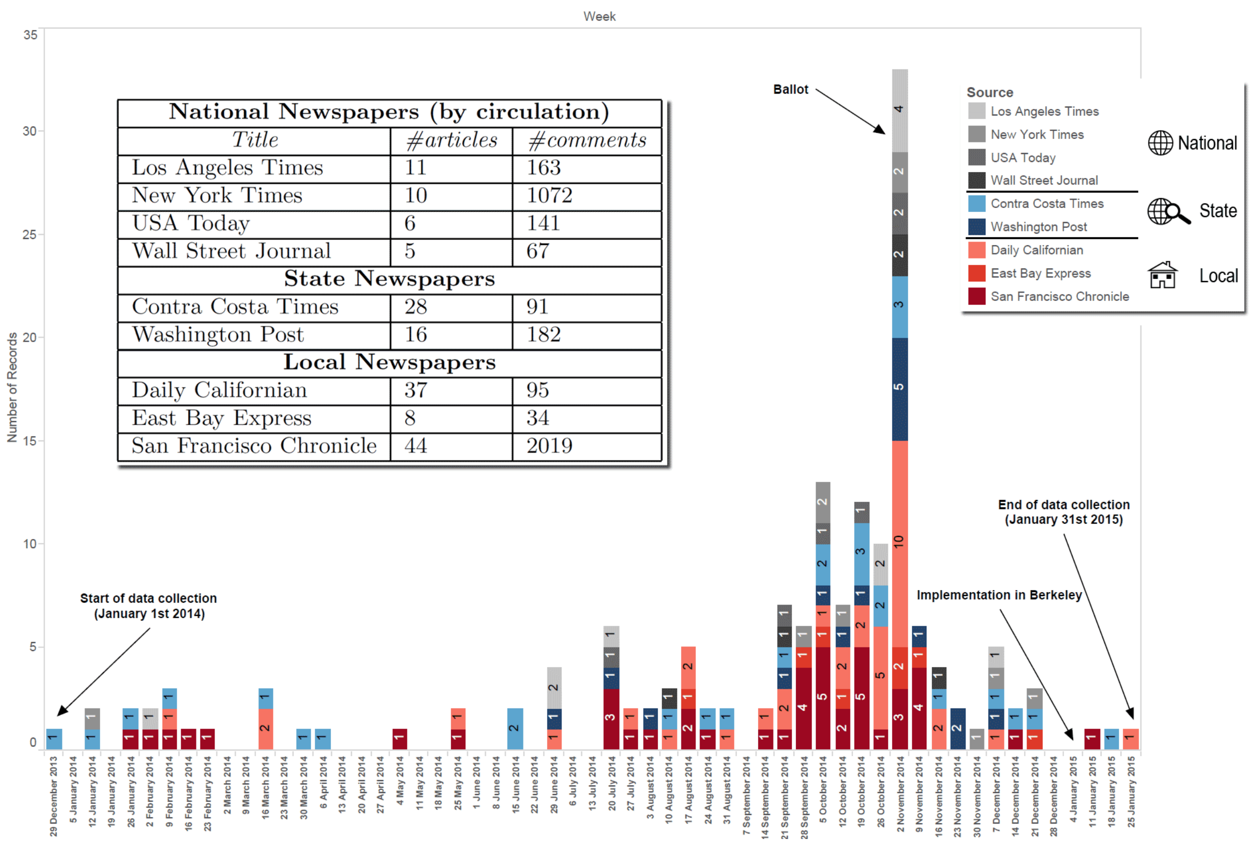
(*Berkeley* OR *San Francisco*) AND *tax* AND (*soda* OR *sweetened beverage* OR *sugary drink*)

OR

(*Berkeley* AND *measure D*)

OR

(*San Francisco* AND *Proposition E*)



National Newspapers (by circulation)		
Title	*#articles*	*#comments*
Los Angeles Times	11	163
New York Times	10	1072
USA Today	6	141
Wall Street Journal	5	67
State Newspapers		
Contra Costa Times	28	91
Washington Post	16	182
Local Newspapers		
Daily Californian	37	95
East Bay Express	8	34
San Francisco Chronicle	44	2019

Figure 3.2: Collected news articles by source and time. The inset summarizes selected newspapers with the number of articles (as found per selection criteria in Box 1) and corresponding comments.

First, we applied the same search criteria which we used to collect news articles about Zika virus in chapter 2. The search criteria was applied via the LexisNexis database, which has a wide reach, particularly of American content. Using this database as the first step is a common approach [110]. This resulted in including the *Contra Costa Times*, *Los Angeles Times*, *New York Times*, and *the Washington Post*.

Second, we repeated the search criteria on each of the 5 largest daily newspapers in the USA (measured by the 2013 combined circulation and online viewing data compiled by the Alliance for Audited Media) via their own search facility. This resulted in adding *USA TODAY* and the *Wall Street Journal*. Third, we repeated the search criteria for newspapers that had a significant readership in either Berkeley or San Francisco. This resulted in including the *Daily Californian* (local Berkeley newspapers), the *East Bay express* and the *San Francisco Chronicle*. Finally, we also applied the search criteria to the top five English-language newspapers outside USA, by circulation figures. None were retained for the analysis as our search criteria did not find enough documents in these newspapers.

3.3.2 Data cleaning and wrangling

Each document was separated into the news article and the readers' comments after removing all the additional information (e.g., advertisements, links to other articles) which were not part of the article itself. Meta-data about the articles was kept in a separate database and contained the article's title, author(s), publication date, newspaper, type of newspaper (i.e., local/state/national), number of readers' comments, and search terms that led to finding the article.

As there were only 165 articles following 9 different formats, cleaning of the article and preparation of the database was done manually, rather than investing in developing cleaning scripts tailored to each newspapers' format. In contrast to the articles, having 3,864 comments made it necessary to process them using scripts. Writing such cleaning scripts can be time consuming, partly because of the wide differences in functionality and formats across newspapers (Table 3.1).

Table 3.1: Differences in comments across newspapers

	Comment functionality					
Newspaper	*Likes*	*Dislikes*	*Reply*	*Full date*	*Comment structure*	*Encoding*
Contra Costa Times	No	No	Yes	No	Set spaces	Text
Daily Californian	Yes	No	Yes	No	Set spaces	Text
East Bay Express	Yes	Yes	No	Yes	N/A	Text
Los Angeles Times	Yes	Yes	Yes	No	Depends on people's use of @	Text
New York Times	Yes	No	Yes	Yes	HTML Structure	HTML
San Francisco C.	Yes	No	Yes	No	Depends on people's use of @	Text
USA Today	Yes	No	Yes	Yes	Set spaces	Text
Wall Street Journal	No	No	Yes	Yes	HTML Structure	HTML
Washington Post	Yes	No	Yes	Yes	HTML Structure	HTML

3.4 Applying Text analytics to the corpus

3.4.1 Solutions most readily available to policymakers

Policy makers can analyze news articles and associated reader comments using well established software such as `Jigsaw` and `IN-SPIRE` which policy makers usually have access to. More functionalities could be obtained from more specialized or research software (e.g., Luminoso from the MIT Media lab [141] or TopicNets [64] from the university of California.

Documents are typically coded for themes. Both `Jigsaw` and `IN-SPIRE` [157] allow themes to emerge (through clustering), as illustrated in Figure 3.3. The height of each theme indicates the number of documents in it, while the words above the theme show the main keywords used by the algorithm to define that theme. The distance between themes indicates how they relate. In this example, it is immediately apparent that there were three broad categories: elections and ballots (left), health (center), company regulations (right). The specific themes within the last category include company sales (which may get impacted by the tax) and changes in can sizes (to compensate for the tax).

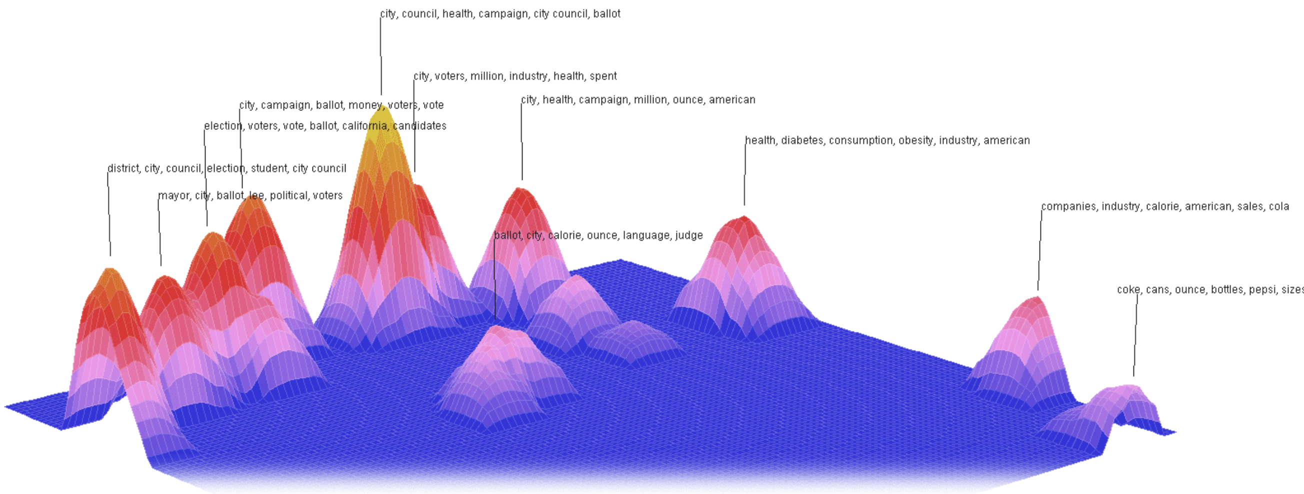


Figure 3.3: Using the Galaxy view from `IN-SPIRE` [157] on the articles.

When trying to assess public opinion, one may seek to examine the context in which specific words are used. An example of a motivating question would be: ‘what do people say about tax?’ One way to do this is to build a word tree using `Jigsaw`. Figure 3.7 shows such a word tree using readers’ comments, and several of the main arguments already appear: some consumers see it as an attack against their freedom to eat a wide range of foods (e.g., a ‘regressive sin tax’ along the lines of taxing cookies or ‘everything that can kill you’), have doubts about the use of proceeds (e.g., ‘revenue to fund other projects or even their own generous pay raises’), fear a disproportional impact on the poor, or even on jobs (‘corporations absorb the hit and reduce jobs’). A similar word tree built on the news articles (provided at https://osf.io/3x6av/), rather than the readers’ comments, depicts a different picture with benefits on childhood obesity and funding health programs more prominently featured.

In addition to this, if a policy maker is interested in understanding the correlation between different events, he can generate a correlation graph which is a visual symmetric matrix which shoes the correlation between the themes as shown in the figure 3.4. Figure 3.5 shows how the themes in the SSB case study emerged over time along with their importance. One of the best ways to get deeper insights is to use Facets in `IN-SPIRE`. Figure 3.6-(a) shows terms which are

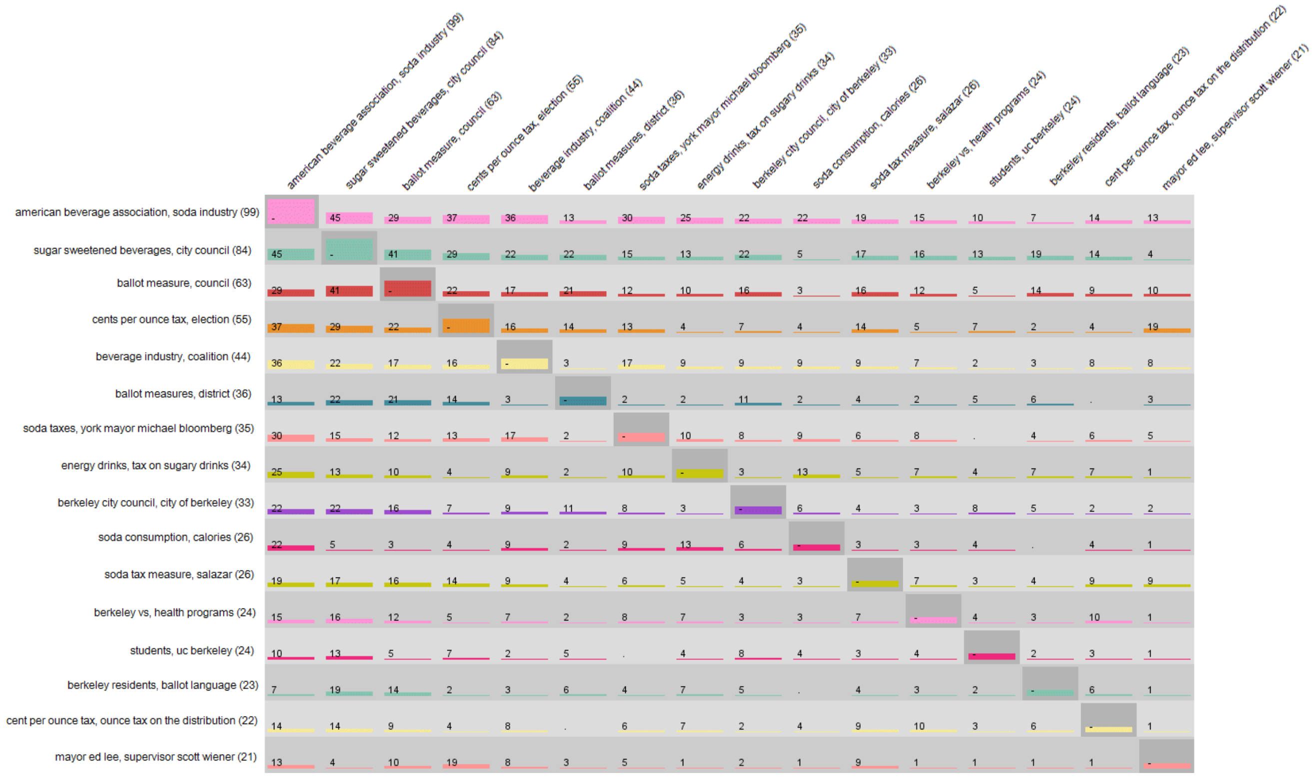


Figure 3.4: Finding correlation between themes.

contributing to negative sentiment, ranked by contribution, figure 3.6-(b) shows a terms which are contributing to positive sentiment, ranked by contribution. figure 3.6-(c) shows all the 165 themes extracted from the corpus. Lastly, figure 3.6-(d) shows the clustering of documents based on the theme using IN-SPIRE's galaxy view.

Finally, a policymaker may be interested in knowing who is behind certain arguments, or how organizations are associated. This can be achieved by entity tracking. Jigsaw automatically processes the documents to identify organizations, persons, locations, and other types of entities. Entity tracking can be done at the micro-level by following links (displayed as lines in Figure 3.8 top): for example, one can start with the American Beverage Association, pick a document in

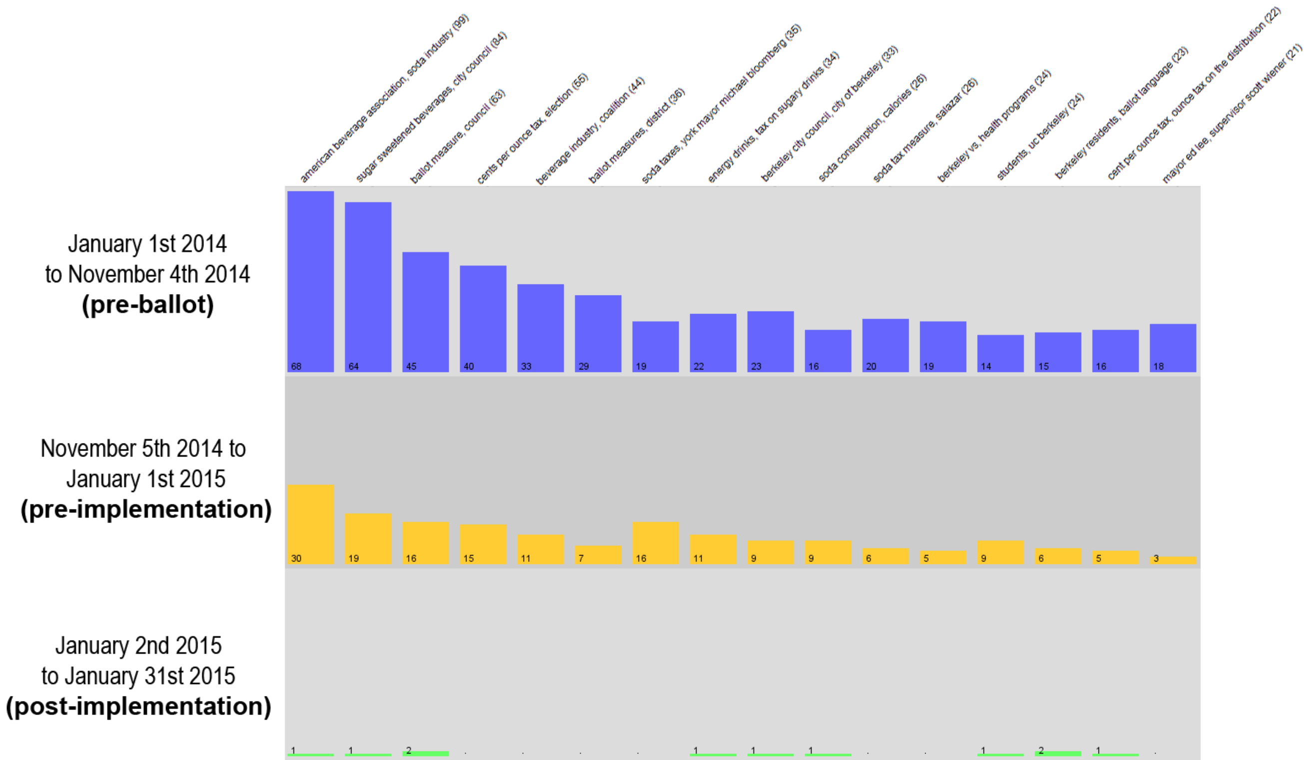


Figure 3.5: Emergence of themes in the articles over time.

which it is mentioned, and see what else is being mentioned. Alternatively, it can be done at the macro-level by finding the entities that co-appear the most with the American Beverage Association (e.g., showing that African-Americans are particularly co-mentioned), across all documents (Figure 3.8)

3.4.2 Advanced solutions

There is still quality of analysis but less accessible to policy makers through off-the-shelf software. Text summarization is of particular interest when policy- makers are faced with a large

Figure 3.6: Sentiment analysis and Theme based clustering using IN-SPIRE.

corpus that they need to quickly condense. We implemented the (graph-based ranking) LexRank algorithm [111] [33]. One issue with this algorithm is that policymakers cannot simply pass it the text and get a summary: they need to choose the value of a parameter, which has a large impact on results. For example, one value can produce an irrelevant summary ("the issue with Fructose is way it is metabolized only by the liver") while another value produces a very relevant one ("the proposed law benefits San Francisco bureaucrats like Scott Wiener who would like to get their hands on that expected $30 million a year by taxing"). By carefully choosing the right value, it is possible to generate summaries and compare how they change depending on the source (news vs readers' comments) or time (pre-ballot, post-ballot, etc.). The temporal difference is clear. For example, the readers' comments post-implementation are summarized as "While some local businesses have felt the effects of the citywide ordinance, sales of sugary drinks sold on campus have not and will not be affected because the UC system is not bound by city laws." In contrast, their comments

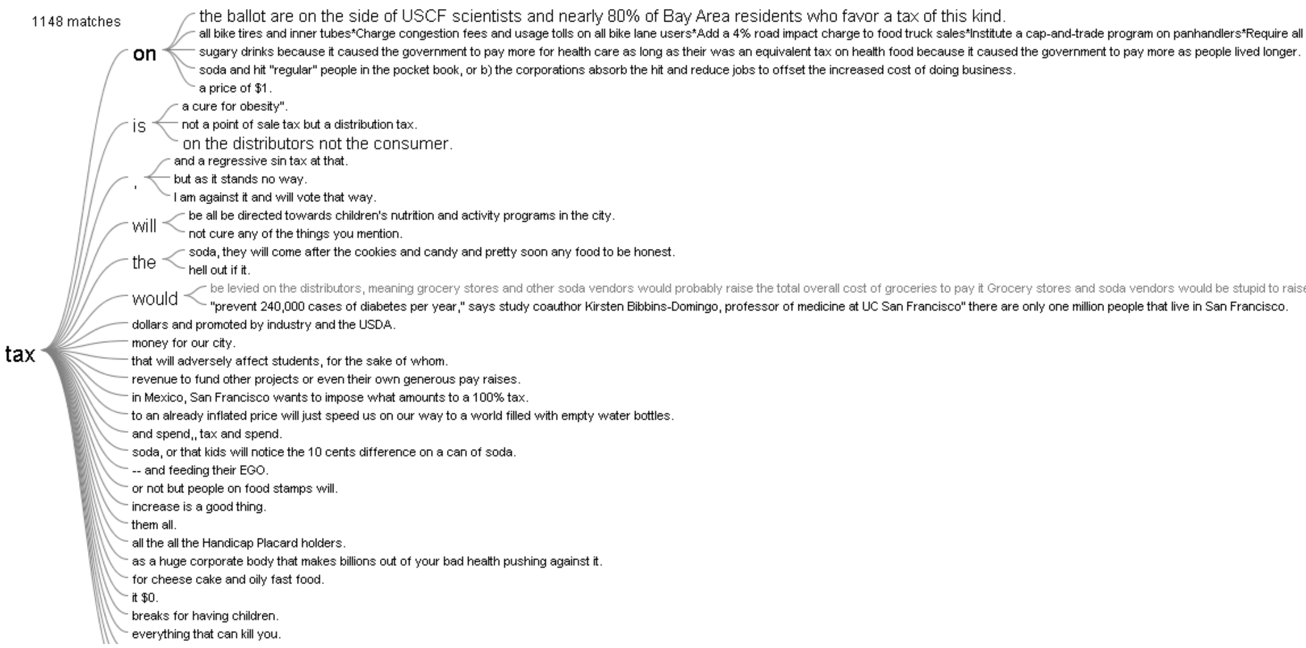


Figure 3.7: Finding readers' arguments that followed the word 'tax' using `Jigsaw`. The full-sized figure is available at `https://osf.io/3x6av/` together with the wordtree based on the articles.

pre-implementation were (perhaps sarcastically) summarized as "If this passes, please continue on and tax red meats, ice cream, donuts, fast food [etc.]". The difference between the article summary and the readers' comments summary echoes observations from word trees (Figure 3.7).

3.4.3 Discussion

In this chapter, SSB taxes were used as a guiding example to detail the process of collecting, cleaning and analyzing data. Off-the-shelf software that policy makers can use to frame interventions were emphasized. We noticed that creation of corpus, preparing the data is time consuming like qualitative analysis. But text analytics and qualitative analysis do not scale in the same way. In text analytics, time is spent in writing scripts (e.g., to collect or clean data), and the time it takes is proportional to the number of different data sources (e.g., one has to decode the format specific to

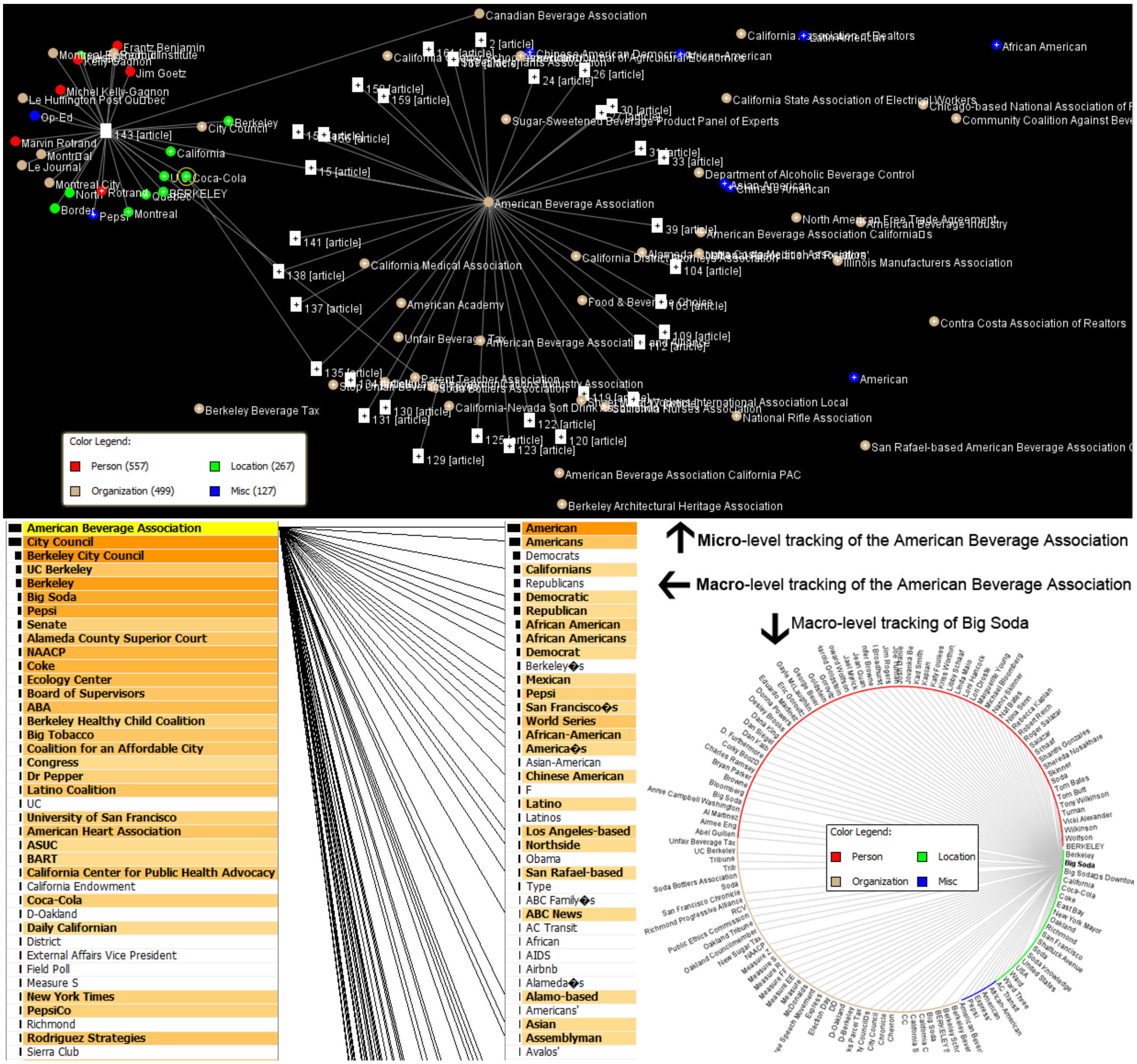


Figure 3.8: Using entity tracking in `Jigsaw` on the articles.

each newspaper). Passed this set-up cost, the cost of analyzing one additional article is negligible. In qualitative analyses, substantial time would be spent in developing the coding framework but analyzing each additional article will also require a small amount of additional time.

The off-the-shelf solutions that we presented have in common that they see all documents as being equally relevant. However, this is not the case in reality. For example, the San Francisco

Chronicle (which had the most articles per our search criteria) also had articles of which the SSB tax was not the subject; instead, the tax may have been briefly mentioned as part of a subject's past records of political endorsements. This could lead to finding irrelevant themes or entities that are not truly connected. Consequently, a more accurate analysis would have to ensure that only relevant parts of the article (if any) are used. Similarly, when assessing public opinions via readers' comments, we need to ensure that the article they comment on strongly relates to the SSB tax.

While emphasizing on a generic procedure to perform text analytics, we note that there are a few limitations affecting the results specific to the guiding example. First, our search procedure cannot claim to have found all articles relevant to the SSB tax debate in California. While it is common to use only one database for text analytics (e.g., [69]), we searched within the LexisNexis database as well as daily news- papers with a large readership either locally or nationally, and the top 5 English- language newspapers outside the USA. This procedure is skewed towards large newspapers, and could be complemented by other online databases such as Access World News (http://infoweb.newsbank.com). In addition, using the largest newspapers does not guarantee that their articles have been at the core of the debates.

CHAPTER 4

ANALYZING COMMUNICATIONS BETWEEN STUDENTS IN A MOOC

The previous chapter discussed how Automatic Text Summarization techniques, and text visualization using off-the-shelf software can reduce human effort in understanding large collections of text files. This chapter focuses on analyzing learners' interactions using network analysis techniques. Analyzing these interactions can be crucial in understanding high attrition rates in MOOCs. We continue our analysis on the same data set and use text classification to avoid manual categorization of students' discussions in chapter 5. Our scripts can be accessed at https://osf.io/dexnd/.

All of this chapter is currently under review following minor revisions as part of the following article:

- Andrew Tawfik, Todd D Reeves, Amy E Stich, Anila Gill, Chenda Hong, Joseph McDade, **Venkata Sai Pillutla**, Xiaoshu Zhou, Philippe J. Giabbanelli. The Nature and Level of Learner-Learner Interaction in a Large-, Open-Enrollment Chemistry Course. In *Journal of Computing in Higher Education*.

The journal's acceptance rate is 15%. My contributions consisted of the social network analysis detailed in this chapter. Analyses conducted by co-authors are not included in this thesis. Data was collected by co-authors.

4.1 Introduction

Similarly to other online courses, Massive Open Online Courses (MOOCs) often rely on learner interaction as a mechanism to promote learning. However, little is known at present about learner interaction in these recently-popularized informal learning environments. Online learning environments use connected network technologies (e.g., discussion boards) to facilitate meaningful interactions [59]. While one might assume that the learners will learn comparably to their counterparts formally enrolled in higher learning institutions, extremely high MOOCs attrition rates are also now well evidenced [88]. Two main strategies to understand the learners' experience include the Interaction Analysis Model (IAM) [66], which categorizes their interaction, and social network analysis (SNA), which focuses on the relation between dynamics (i.e. who interacts with whom over time) and learning outcomes. This chapter focuses on SNA while the next chapter deals with the IAM.

SNA has been used in many occasions to study MOOCs [46]. For example, it allowed Goggins *et al.* to find that the social interactions in a MOOC changed over time, and in particular, they tend to sparsify [58]. In Gillani and Eynon's [56] study of one Coursera MOOC, they also found sparsification of topical discussion forums over time. In addition, they found that the MOOC network featured multiple small, dense networks rather than a single connected network. Other studies have tried to link SNA measures such as centrality to course performance, with mixed results [31].

Our study relies on data from a specific Science, Technology, Engineering, and Mathematics (STEM) MOOC offered through the prominent MOOC provider, Coursera. The MOOC is simply titled "Chemistry". In this class, participants were asked to collaborate on a discussion board that corresponded with the module topic. Students were also provided two additional discussion boards (General Discussion, Study Groups) which did not correspond with a given week, but were open ended. The data was downloaded by individual discussion posts. The analytic data set comprised 274 unique students and three unique instructors/Teaching Assistants (staff). The data encompasses one complete session of the class with over 992 posts in 7 topic areas across all forums. These interactions between students can be seen as a network of student nodes. Each time they interact, a connection between is made and represented by edges.

This chapter is organized as follows. In section 4.1.1, we discuss what network metrics (introduced in Chapter 2) mean in the context of MOOC. Then, we apply these metrics to our MOOC in section 4.1.2. Finally, section 6.6 contextualizes the result in terms of learner-learner interaction, and suggests complementary analyses for future work.

4.1.1 Metrics in context: what they mean in a MOOC

It is common to use Social Network Analysis (SNA) when studying the interactions of a group of people [44]. To apply SNA, we need to represent the interactions in a MOOC as a graph. The graph is typically undirected, represents persons as nodes (e.g., anybody who posted a topic or made a comment/reply), while their interactions form connections (e.g., when a person replies to another in a topic thread) represented as edges (Figure 4.1) [44].

With the graph created we can perform SNA by measuring various aspects of the graph. The literature on SNA applied to MOOC includes several case studies using the notion of centrality (section 2.2). For instance: Rabbany et al. [119] built *Meerkut-ED*, a tool for assessing student participation. They used degree centrality to observe the relative importance and influence of students in the MOOC discussions. They generated a student interaction graph at different timestamps using a radial layout (as in section 2.2) placing more central students towards the center to monitor the change in students' behavior throughout the course. García-Saiz et al. [44] proposed a tool called Elearning Web Miner using different centrality measures such as Degree, Betweeness

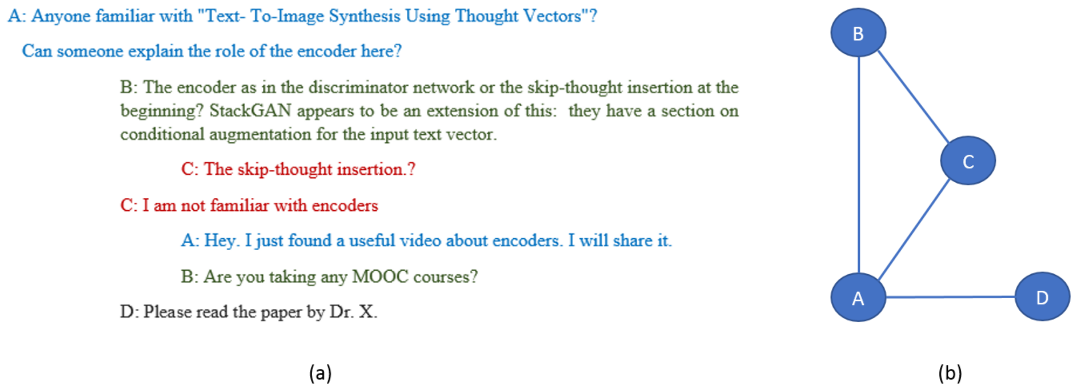


Figure 4.1: (a) Discussion Thread showing the interaction between the users: A,B,C, and D. (b) The corresponding un-directed graph.

centralities on a graph generated based on interactions between instructor and students. Dowell et al. [31] used network properties such as degree, closeness, betweenness, and *Coh-Metrix* [61] to investigate relationships between learners' discourse and social centrality. They showed that learners who have a conversational style discourse with simpler syntactic structures occupy a more central position in the network. There have also been works employing SNA metrics in the context of MOOC, but focused on the *field* of MOOC research rather than MOOCs themselves. For example, Gasevic et al. [46] identified the association of different factors with the success rate of acceptance of proposals related to MOOCs on citation and co-authoring networks.

In addition to (relatively) straightforward centrality metrics such as degree or betweenness, Yang et al. [159] employed advanced metrics including the *authority score* (corresponds to the importance of information stored in that node) and the *hub score*(corresponds to the quality of nodes' links) [85] to explore student dropout behavior in MOOCs. Their results show that students with authority scores that are one standard deviation higher than the average are 33% more likely to drop out of the course.

4.1.2 Methodology

As stated in Section 4.1.1, we built several graphs to analyze the social network and the interactions between participants. We view the graphs at two different levels. The measures employed in our analysis are at both the node- (e.g., centrality indices) and network-level (e.g., density and average path length introduced in Definitions 2.2.7 and2.2.8 respectively). In other words, we measure the dynamics of individual participants (i.e. nodes) and the community as a whole (i.e. network).

4.1.2.1 Macro level: the whole community

In this section, we gain a basic understanding of the graph created from the MOOC using SNA. We do this first with coming up with graphs of each individual topic and measuring how it changes.

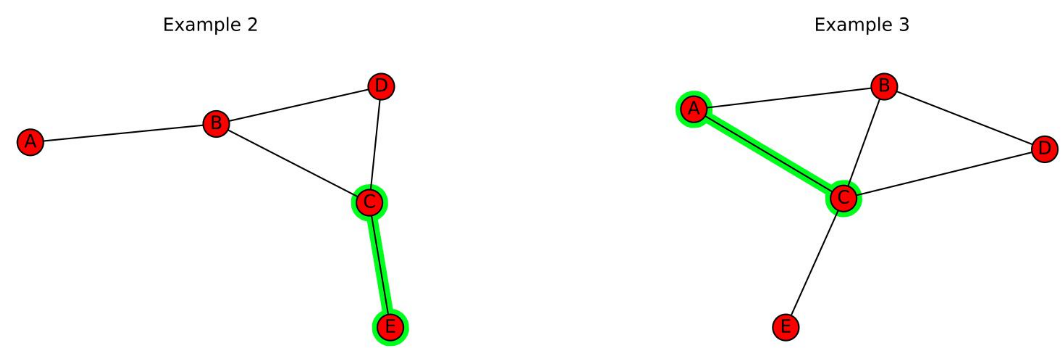


Figure 4.2: Examples demonstrating graph edges and nodes.

In Figure 4.2 we can notice how node C forms a connection with node E and later with node A. The graph is becoming more connected as the community becomes more collaborative. In SNA terms the network is becoming denser and the average shortest path length is decreasing. To find and visualize this data, we create Table 4.1 showing the changes.

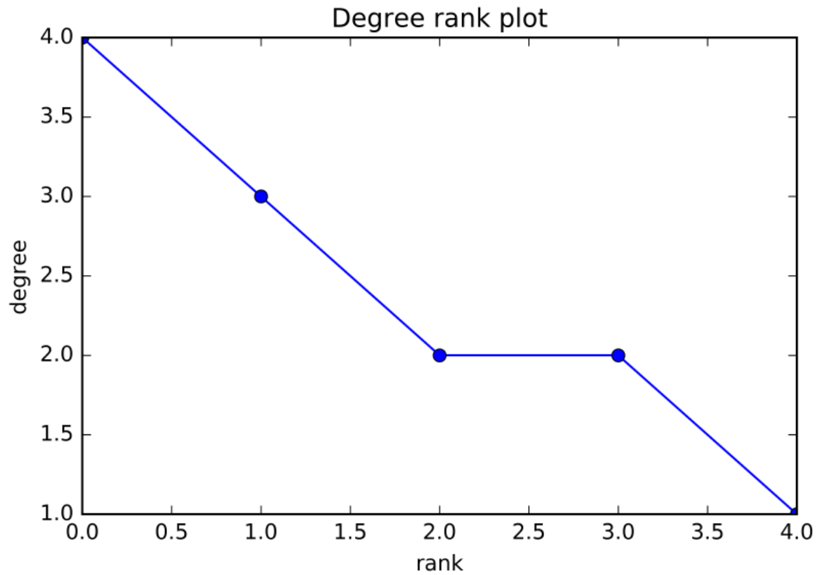


Figure 4.3: Figure demonstrating degree-rank.

	Ex: 1	**Ex: 2**	**Ex: 3**
Average Shortest Path	1.067	1.6	1.4
Nodes	5.0	5.0	5.0
Edges	4.0	5.0	6.0
Average Degree	1.2	1.2	2.0
Density	0.4	0.5	0.6

Table 4.1: Different metrics for the example graph.

We also visualize the degree rank distribution of the network with a degree rank plot. On the Y axis we have the degree of the node (higher is more important), on the X axis is the rank or importance of the node (lower being more important). The network we are most interested in is that of user interactions. A node in the network corresponds to a person. A link between two persons exists when one person directly answered a post made by another user. For example Joseph starts a post with the title “Any one started working on the project?” then John answers his question by posting “Yes.. I did”. Sriram then answers with his post “nailed it”. Then the network structure is $Joseph \leftarrow John \leftarrow Sriram.$

4.1.2.2 Micro level: individual participants

In Section 4.1.4 we look for trends in centrality. In our example graph we found the important players and Micsaw node C make more connections. At the start node B was the most connected participant and later we saw node C make more connections and even become more connected than node B. This is visualized using a parallel coordinates plot. Along the Y axis is the measure of normalized degree centrality. The X axis is various example graphs (e.g., Example 1, Example 2, Example 3). Each line is a node changing position as connections change from each example network.

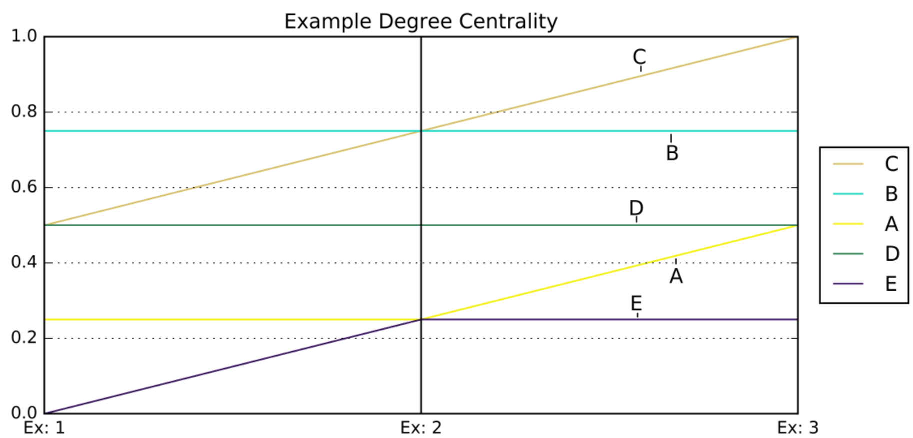


Figure 4.4: Figure demonstrating degree-centrality of the above example.

4.1.3 Comparing measures for each module

In this section the comparison of various metrics are shown. These metrics include: average shortest path, number of nodes, number of edges, average degree and density of the different topics. Table 2 shows the metrics for each individual topic. Figures 4.5 and 4.6 show the centrality degree distribution for individual nodes from Topic 1 to Topic 7 cumulatively.

Discussion board	Nodes	Edges	Density	Average degree of nodes	Average Distance
Topical (weekly)	-	-	-	-	-
Topic 1	76	115	0.02	1.18	3.10
Topic 2	33	52	0.05	1.45	2.96
Topic 3	11	12	0.11	0.82	3.02
Topic 4	27	33	0.05	1.30	2.24
Topic 5	17	16	0.06	0.88	3.53
Topic 6	17	14	0.05	0.35	1.84
Topic 7	18	17	0.06	0.83	2.27
General Discussion	147	180	0.01	0.64	5.02
Study Groups	47	78	0.04	1.60	3.12

Table 4.2: Comparison of measures for Topic 1-7

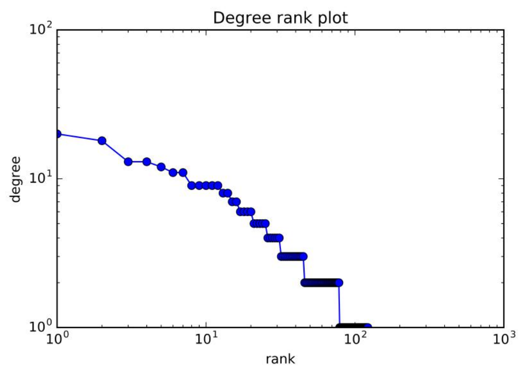


Figure 4.5: Degree distribution on a log-log scale for topics 1-7.

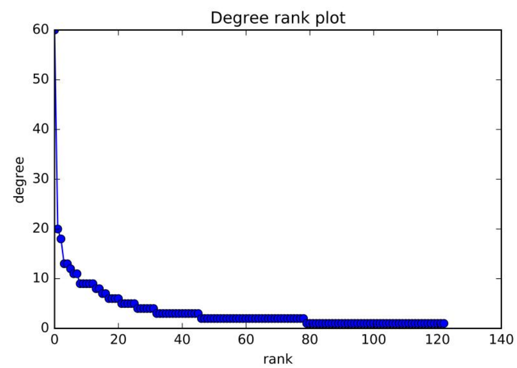
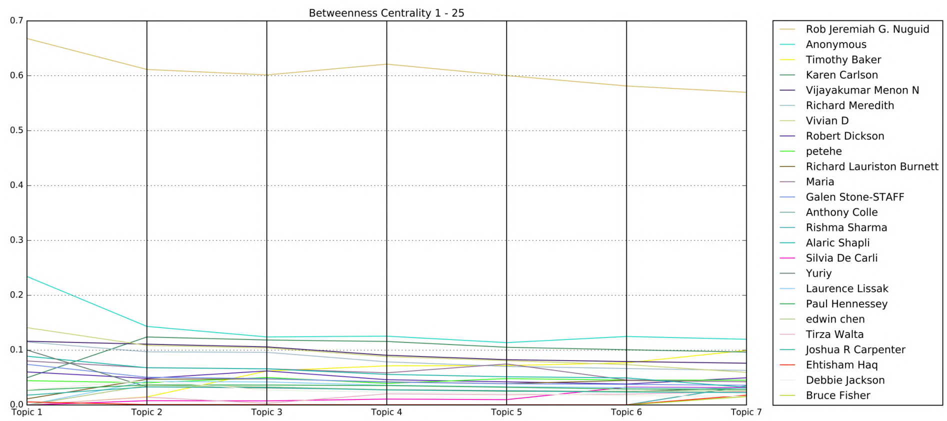


Figure 4.6: Degree distribution on a linear scale for topics 1-7.

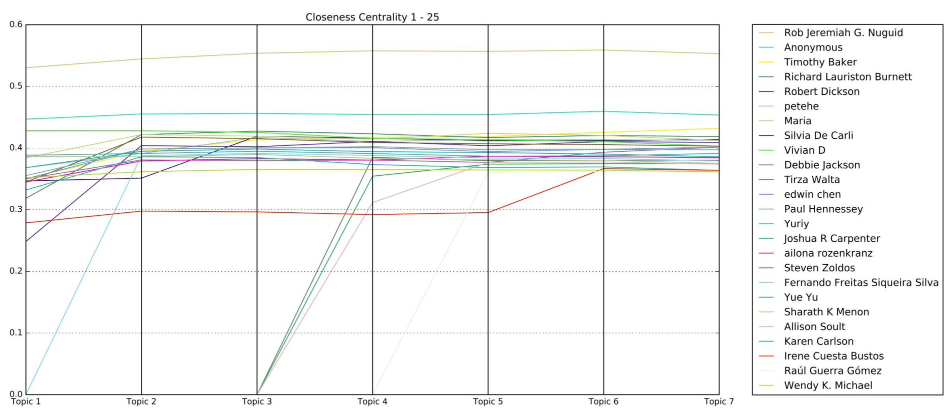
4.1.4 Trends in centrality

This section details the change in centrality of the nodes as the course elapses. To understand this shift in the nodes, degree centrality, betweeness centrality, closeness centrality were computed for all the individual topics. These centralities of the nodes were plotted against individual topics using *parallel coordinates* in figures 4.7a, 4.7b and 4.7c.

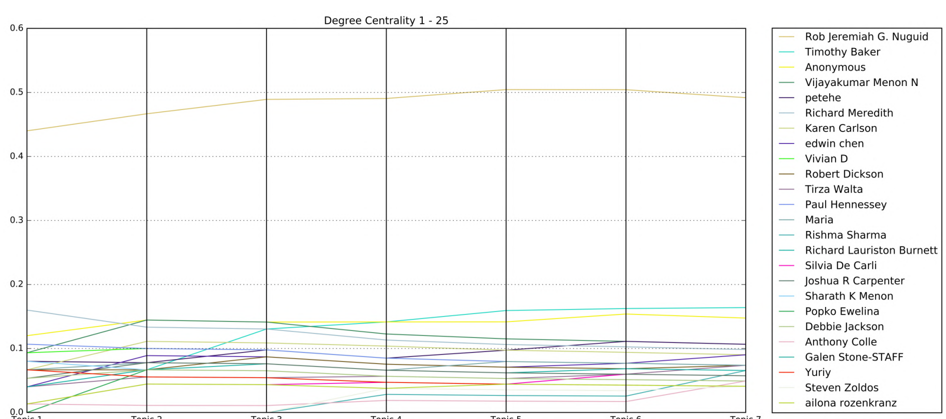
Results are shown in Figure 4.7 for the top 25 nodes by centrality. Others are not shown for readability, and because their centrality became so small that changes can be due to noise. Results



(a) Shifts in Betweeness-centrality of 1-25 nodes



(b) Shifts in Closeness-centrality of 1-25 nodes



(c) Shifts in Degree-centrality of 1-25 nodes

Figure 4.7: Shifts in centrality of 1-25 nodes.

suggest that the role of the most central participants changed little over time: those who answered multiple posts essentially persisted in this behavior (Figure 4.7c).

4.2 Discussion

Both content analysis and social network analysis indicate a marked decrease in discussion board activity as the course elapsed. One consistent finding of research on other MOOCs is that these courses are characterized by high dropout rates, as much as 90% [72]. When exploring the macro level of the MOOC some important discoveries are made. The level of participation drops off very quickly. Starting with the General Discussion forum, there are 147 participants. In the final topic of the MOOC, Topic 7, there is only 18 remaining participants. Only approximately 12% of participants finished the MOOC. There is also a problem of low density. Density relating to the amount of edges between the nodes implies that this MOOC had low levels of discussion. Finally, we see the average degree and average shortest path remain low throughout the various topics. After plotting the degree distribution, it is clear to see that there are only a few highly connected nodes. This may imply that one or two participants are answering questions for many of the other participants instead of encouraging discussion.

At the micro level we tracked the trends in centrality. All topics were added cumulatively from Topic 1 through Topic 7. In order to better analyze the data, it is visualized in Figure 4.6. The SNA from the current study shows that the number of learner interactions quickly becomes more sparse, and the little interaction that remains is heavily dependent on a few highly-engaged learners.

The visualization only includes the top 25 participants as the data gets very noisy as more are included. It is clear from the visualization that there is one very well connected participant, Rob Jeremiah G. Nuguid. This seems to confirm the results of the macro level analysis. SNA also indicated that the number of participants who were actively participating quickly declined after the early modules. When learners did interact, the low average number of interactions shows that they rarely interact with more than one other classmate. Additionally, the SNA found that the interaction patterns were different in the latter part of the course when compared with the early modules. The students that did persist contributed to a greater degree as the course progressed.

CHAPTER 5

CLASSIFYING STUDENTS' DISCUSSIONS IN A MOOC FORUM

The previous chapter explained the dynamics of interactions between students in a MOOC course using Social Network Analysis and Content Analysis with the help of IAM framework. But, manually categorizing student discussions into IAM framework categories is an arduous task. Supervised learning is the machine learning task of determining a function from labeled training data to automatically categorize new data. We briefly discussed text classification in section 2.5.1. In this chapter, we will explain how text classification can solve the problem of automatically categorizing students' interactions. Chapter 4 and chapter 5 can synergistically improve online education by providing a means to track students' performance. Our scripts we wrote to build the classification model can be accessed at *https://osf.io/68feu/*.

All of this chapter has been submitted as the following article:

- **Venkata Sai Pillutla**, Andrew Tawfik, Philippe J Giabbanelli. Automatically identifying the phases of learner-learner interaction by applying data mining to a large-, open-enrollement course. In *Computers in Human Behavior*.

The data used in this chapter is the same as was collected by co-authors of the previous chapter.

5.1 Introduction

Text classification has met with significant success in real world text-based learning tasks. In Machine learning, classification is the problem of identifying to which set of categories a new observation belongs to. Text Classification was briefly discussed in section 2.5.1. A more detailed explanation about the usage of different machine learning algorithms for text categorization and can be found in [1]. Text classification started gaining importance in improving education, especially in an online setting (MOOCs).

In this chapter, we present how text classification can help instructors to assess learners' performance based on their interactions. The main aim of this chapter is to provide an automatic means to assess where learners stand in terms of the IAM Framework, rather than the current resource-intensive approach of manually reading and categorizing all of the learners' activities.

We will present recent work on classification of student's posts and their shortcomings in section 5.2. In section 5.3 we introduce a popular framework to understand learner interaction: the IAM Framework.Finally, we present our automatic category identification approach in section 5.4.

5.2 Related work

Rossi and Gnawali [128] proposed a language independent text classification model to categorize text based on the discussion threads in coursera MOOC forums. They classified the discussions into social/small talk, open ended topics, (un)resolved close ended problems (e.g. assignments), course logistics, etc. The main aim of their model is to help the course instructor notice and solve the unresolved problems and make the best use of his time. They defined and tested language independent features for the supervised classification of threads. Since their goal was to build a language independent model, they cannot use an n-gram model as an input to their classifier. They considered features such as no. of views, no. of votes, No. of posts in a thread, no. of unique users in a thread, etc. They peformed the classification via a Support Vector Machine (SVM) classifier with linear kernel and obtained a precision and recall of 0.686 and 0.700 respectively.

Liu et al. [95] proposed another classifier to reduce the workload on course instructor. They trained the classifier on forum data from the Scala MOOC given by the EPFL in 2014. They categorized user posts into the target categories: Question, answer, clarification request, clarification, positive feedback, negative feedback, off-task (Spam or misplaced text). The basic set of features chosen by Liu et al. include Number of words in the post, number of sentences, number of spelling errors, number of question marks, no. of relevant words, etc. A set of relevant words were selected from the lecture slides and posts. Presence of such words in user posts was counted as a feature to capture the relevance of a post to course material.

Agrawal and Leonard [2] proposed an pipeline to mine confusion from the user posts. A post is categorized as 'confusing' if its author explicitly asks for clarification or implictly reveals a gap in his understanding of some concept. Identifying confused posts can reduce the effort of the instructor and help him answer those relevant forum posts. The other target classes in their model are 'neutral' and 'knowledgeable'. They built their model using ngrams and a combination of n-grams features and non-n-gram features such as no. of upvotes, no. of reads, time stamp, etc. Their evaluation reveals that there is no significant increase in the precision and recall when additional features were added to the n-gram feature space.

These three models discussed above had their limitations: The target classes used in their models, do not cover all possible forum interactions. For example, Some students just share their thoughts without the intent of asking any questions. This case isn't covered. Some students tend to re-write or cite the question before answering it. These kind of posts may be misclassified as questions instead of answers. Hence, we chose to use the Interaction Analysis Model (IAM) framework which can categorize the learner interactions into a broader set of categories. IAM Framework was being used in various studies [156, 63, 36, 146, 129] involving analysis of discussions.

5.3 Interaction Analysis Model framework

A prominent framework to conceptualize learner interaction is the interaction analysis model (IAM) [66]. IAM posits that the process of learner-learner interaction advances through five phases. These phases of interaction reside along a continuum that begins with sharing information (Phase 1) and progresses to application of newly constructed knowledge (Phase 5). In Phase 1, individuals share and compare information. In Phase 2, they are able to identify areas of disagreement from their previously shared ideas. Once discrepancies in understanding are identified, Phase 3 describes learner-learner negotiation of meaning as they weigh evidence, identify areas of overlap in their understanding. Phases 4 describes how learners synthesize their agreed upon knowledge. Finally, Phase 5 consist of agreement statements about how to apply their knowledge. In the IAM, each phase also comprises sub-phases. For instance, Phase 1 consists of stating an opinion (P1:A), agreeing with peers (P1:B), providing examples (P1:C), clarifying details (P1:D), and identifying problems (P1:E). Alternatively, Phase 5 includes sub-phases such as summarization of agreements (P5:A), applications of new knowledge (P5:B), and instances of metacognition (P5:C). To date, the IAM framework has been applied to understand learner-learner interaction in relation to formal online higher education. Less attention has been given to interaction in MOOCs. A key contribution of this chapter is to investigate the extent to which the phase of the IAM framework can be inferred from MOOC traces using text mining.

5.4 Automatic category identification

5.4.1 Data pre-processing

In general, pre-processing for classification of text data involves two main steps: (i) ensuring that all classes are balanced such that none will skew the classifier (which applies to all classification problems), and (ii) transforming the text into a feature vector (which is specific to text classification). This section summarizes both in turn.

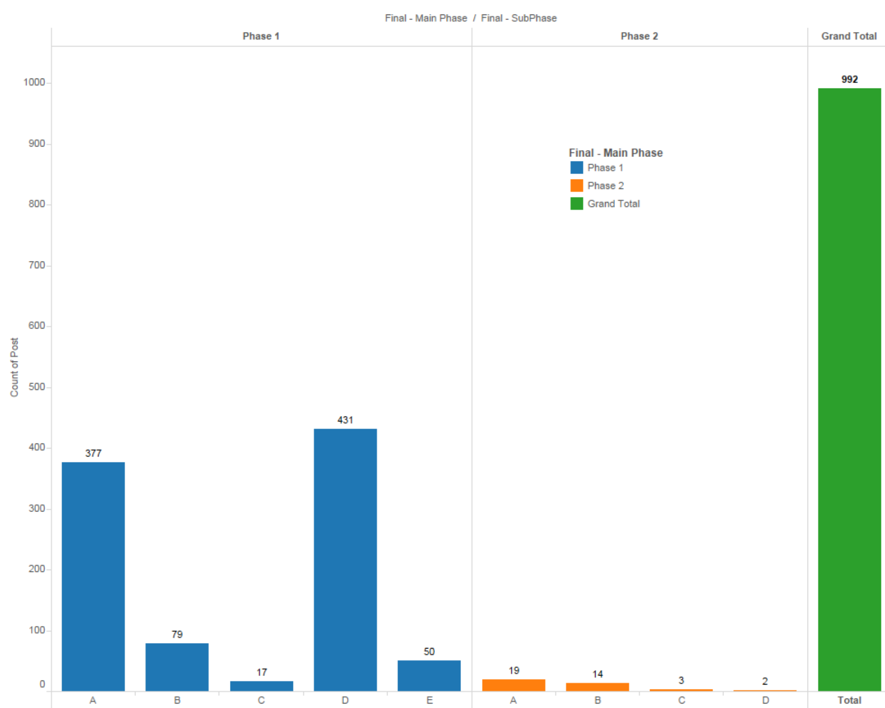


Figure 5.1: Distribution of posts per phase category.

In our data, classes are imbalanced (Figure 5.1). The issue of imbalance has been extensively discussed by Japkowicz [78]. As noted by Poolsawad and colleagues, "on such data, learning classification methods generally perform poorly because the classifier often learns better the majority class" [116]. This problem arises in part because classifiers assume that the training data is balanced and that errors have the same cost [116, 121]. It can be addressed either by changing the error costs, or by balancing classes [107]. Balancing can be achieved by either under-sampling (i.e. taking sub-sample uniformly at random of the majority class) or over-sampling (i.e. creating synthetic examples of the minority class). Phase 2, which has a total of 38 samples across all its sub-phases, cannot be over-sampled hence it was excluded from the analysis. Different approaches were proposed in the literature to over-sample the minority class (e.g., Synthetic minority over-sampling technique [20], and Adaptive synthetic sampling [70]).

In total, sub-phases 1-B, 1-C and 1-E have 146 observations. This is less than the total for either 1-A (n=377) or 1-D (n=431). Thus we merged 1-B, 1-C and 1-E to form a new class: 'other posts'. To achieve a balanced distribution of classes, we over-sampled 'other posts' to 292 posts using Synthetic Minority Over-Sampling Technique (*S.M.O.T.E*) and under-sampled the sub-phases A and D to 292 posts each by selecting a random subset. This introduces randomness in the process, hence datasets were generated multiple times (as detailed in the next section). To sum up, the final target classes for our model are '1-A', '1-D', and 'other posts'. After balancing the classes, there are 292 samples for each class (876 samples in total).

When transforming the text into a feature vector, there are two key parameters: the number of features to retain (*max_features*), and the threshold,*max_df*. The *max_features* parameter specifies how many features(sorted on the word frequencies) to retain. *max_df* specifies the vectorizer to ignore terms that have document frequency lower than the given threshold. The choice of their values has an important impact over the classification results, thus we explored which combinations were optimal. Results are reported in Figure 5.2 and show that $max\_features = 3731$ (i.e. using all features) and $max\_df = 0.7$ yields the best results. In the next section, the data fed to the *TfidfVectorizer* (and then provided to the classifiers) will thus always be transformed with these parameter values.

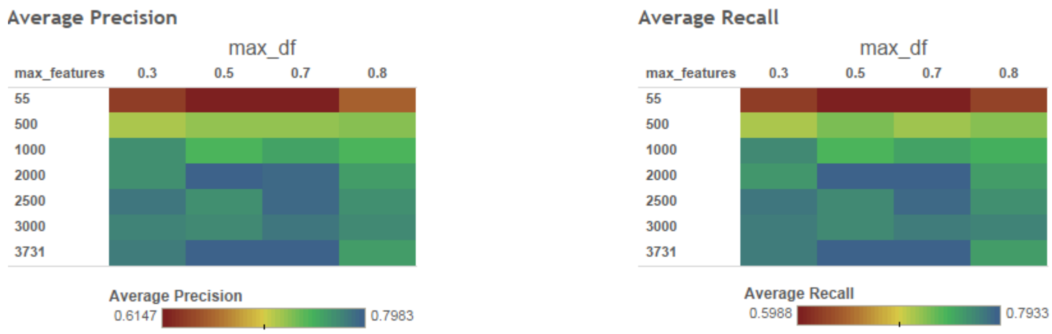


Figure 5.2: Heatmap showing how number of features and the maximum document frequency filter in the *tf-idf* conversion impact the average precision(left) and recall(right). The best values of precision and recall are obtained for $max\_features = 3731$ (i.e. using all features) and $max\_df = 0.7$. The LinearSVC was used as classifier.

5.4.2 Analysis

SVMs with two kernels (Linear and RBF) [22], Decision Tree, and Random Forest classifiers were trained using the data set. Scikit-learn 0.17 was used to build the classifiers [114]. Linear SVC is similar to SVC with linear kernel but, it is implemented using *liblinear* instead of *libsvm*, which gives a wide choice of penalties and loss functions [138]. Radial basis function (RBF) Kernel SVC, Polynomial kernel SVC, etc. can be used when the data points are linearly inseparable [5], but text data, being high dimensional, is often linearly separable [81].

After building the classification model, *precision*, *recall* , *f1 score* and *support* are computed for the individual classes per definitions 2.5.1, 2.5.2 and 2.5.3.

Based on their content, posts were classified into respective IAM categories which correspond to a particular course module (week). 10-fold cross validation is performed on the data [92]. This process repeats 10 times and each time one of the 10 parts is used to test the classifier which is trained on the remaining 9 subsets. Since we over- (via SMOTE) and under- sampled the data to balance classes, we ran the classification models 10 times each and report on the average in table 5.1. The best classifier as reported in the Table is LinearSVC, which provides an average

Classifier	**Parameters**	**Average Precision**	**Average Recall**	**Average F1 Score**
Linear Kernel SVC	kernel = “linear”, class_weight=“balanced”	0.782794029	0.776437692	0.776323357
LinearSVC	class_weight=“balanced”	0.789062036	0.783300561	0.789062036
SVC RBF	kernel = “rbf”, class_weight=“balanced”	0.150452869	0.356640705	0.197958361
Decision Tree Classifier	criterion=‘gini’	0.652521205	0.63692288	0.632572003
Random Forest Classifier	criterion=‘gini’	0.720407507	0.704541624	0.699219153

Table 5.1: MOOC post classification results of 5 classifiers trained on *tf-idf* representation of user posts with $max\_features = 3731$ and $max\_df = 0.7$. Note that there are many additional parameters (as will be further discussed in the case of LinearSVC); parameters not reported in the Table were left to their default value.

precision of 0.79 and an average recall of 0.78. Having optimized the vectorization and selected the best classifier, we now turn to exploring the parameter space for this classifier in order to further improve classification accuracy.

After identifying the threshold *max_df* and *max_features* we performed a parameter sweep (Figure 5.3) to optimize the SVM with the fixed threshold *max_df* as 0.7 and *max_features* = 3731 and identified that the maximum average precision, recall and f1-scores were obtained with the default parameters of SVM on Scikit-learn: ‘kernel’: ‘linear’, ‘C’: 1.0, ‘verbose’: False, ‘probability’: False, ‘degree’: 3, ‘shrinking’: True, ‘max_iter’: -1, ‘decision_function_shape’: None, ‘random_state’: None, ‘tol’: 0.001, ‘cache_size’: 200, ‘coef0’: 0.0, ‘gamma’: ’auto’, ‘class_weight’: ’balanced’.

In our search for the best classifier, and the optimization of the LinearSVC, we repeated experiments 10 times (and each used a 10-fold cross validation). Having established the best one, we now check whether additional experiments are required or if 10 times was sufficient. We use the confidence interval method to compute the number of required runs. Specifically, we computed the confidence interval to give an estimated range within which the true mean average precision

Loss	Penalty	Dual	Cost C	Standard Deviation Precision	Average precision, recall, and F1 score
squared_hinge	l1	False	1	0.0366731	0.740921, 0.738407, 0.729967
			1.5	0.0504734	0.753317, 0.745584, 0.74398
			1.75	0.0449285	0.75225, 0.744062, 0.742787
			2	0.0456787	0.752754, 0.745812, 0.743342
		True	1		
			1.5		
			1.75		
			2		
	l2	False	1	0.0400154	0.78295, 0.776941, 0.775542
			1.5	0.0455334	0.78536, 0.778882, 0.776742
			1.75	0.0402259	0.77683, 0.769626, 0.767256
			2	0.0457716	0.776986, 0.770755, 0.768409
		True	1	0.050498	0.789062, 0.783301, 0.789062
			1.5	0.0410193	0.783534, 0.776297, 0.773962
			1.75	0.047854	0.777839, 0.77026, 0.768265
			2	0.0428265	0.778128, 0.769834, 0.767227
hinge	l1	False	1		
			1.5		
			1.75		
			2		
		True	1		
			1.5		
			1.75		
			2		
	l2	False	1		
			1.5		
			1.75		
			2		
		True	1	0.0434058	0.780625, 0.77537, 0.774138
			1.5		0.775625, 0.771226, 0.769837
			1.75	0.0394477	0.776116, 0.770428, 0.768827
			2	0.0440133	0.784572, 0.779197, 0.777514

Figure 5.3: Parameter sweep to optimize the SVM. Empty cells correspond to combinations that are not technically valid.

is expected to lie. The confidence interval for the average precision is calculated using equation 5.1 [127].

$$CI = \bar{X} \pm t_{n-1,\frac{\alpha}{2}} \times \frac{S}{\sqrt{n}} \tag{5.1}$$

where:

$\bar{X}$ = mean of the output data from the replications

S = standard deviation of the data from the replications

n = number of replications

$t_{n-1,\frac{\alpha}{2}}$ = value from t-distribution with n-1 degree of freedom and significance level $\frac{\alpha}{2}$

In our case, the average precision of the initial 10 replications performed is 0.7890, standard deviation is 0.050. The Confidence Interval from 5.1 is found to be $(0.7484, 0.8218)$ with a confidence of 95%.

$$n = \left(\frac{100St_{n-1,\frac{\alpha}{2}}}{d\bar{X}} \right)^2 \tag{5.2}$$

where:

d = the percentage deviation of the confidence interval about the mean.

We determined the number of replication required $(10.04 \approx 10)$ using equation 5.2 [127] which is obtained by rearranging confidence interval formula. Consequently, 10 simulations were sufficient to report results within a 95% confidence intervals and we did not run additional experiments.

The final average precision, recall, and f1-score percentages of the 10 replications of linear SVC with the default parameters are found to be 78.90, 78.33, and 78.90 respectively.

5.5 Discussion

We built a classification model which can be used to automatically assess where a student stands in terms of the IAM Framework based on their posts. Text data, which is high dimensional, is usually linearly separable [81]. It can be understood from the results in table 5.1 that SVC Linear is giving a better accuracy than the other classifiers. However, accuracy may be improved by getting more training data which we are in lacking in our case

The advantage of using the IAM Framework over target classes from other studies [2, 95, 128] is its ability to capture whether knowledge is constructed within the group by means of the exchanges. It also identifies if individual participants change their understanding of the subject as result of interactions with other participants. Hence, the target classes from the IAM Framework are more effective than the target classes from other studies. While building this model, we only considered uni-gram features rather than engineering new features such as no. of words in the posts, no. of spelling mistakes, etc. As suggested by the study of Agrawal *et al.* [2] we chose only to use n-gram features as the study shows that there isn't any significant increase in the accuracy of the model when additional features are considered.

Our classification model can support many MOOC stakeholders in automatic assessment of students, which is costly to evaluate manually. The model can be used in performing large scale analysis of performance of students participating in MOOC forum discussions, and answer questions such as, (i) are students answering fellow students' questions performing better than others? (ii) Do students who express gap in their understanding, drop out eventually?

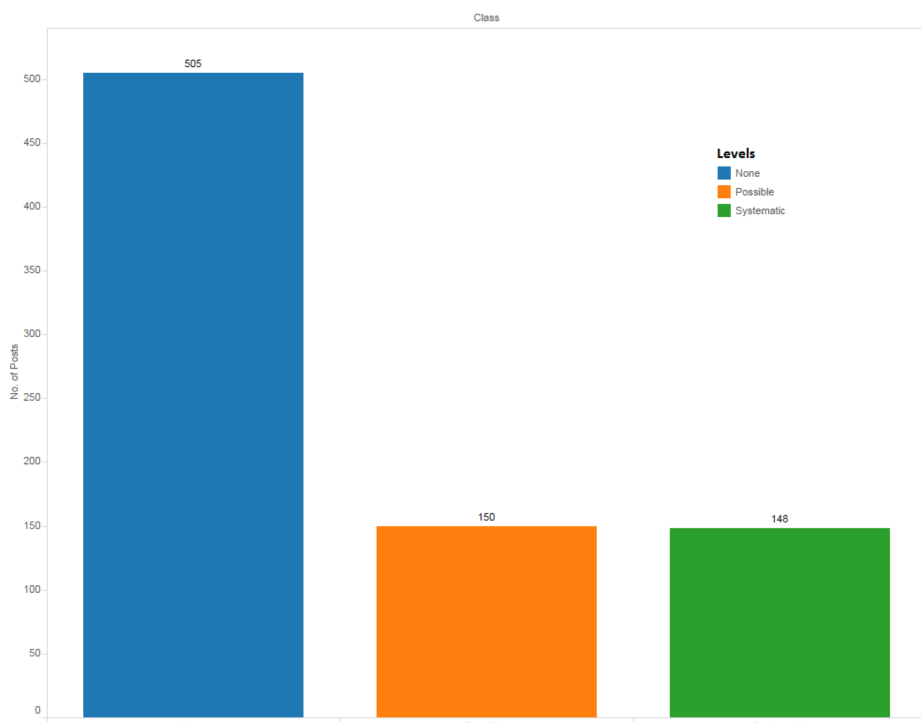


Figure 5.4: Distribution of posts per mis-classification category.

There are some limitations of our work: (i) Since the data set is from a chemistry MOOC course and the classification model built on the uni-grams of chemistry terminology, it cannot

Classification Model	**Average Precision**	**Average Recall**	**Average F1-score**
DecisionTreeClassifier	0.358335423	0.355050505	0.346046078
SVC Linear	0.450877483	0.435151515	0.433928236
SVC RBF	0.14905676	0.352323232	0.201766202
Random Forest	0.439400611	0.403737374	0.400793703

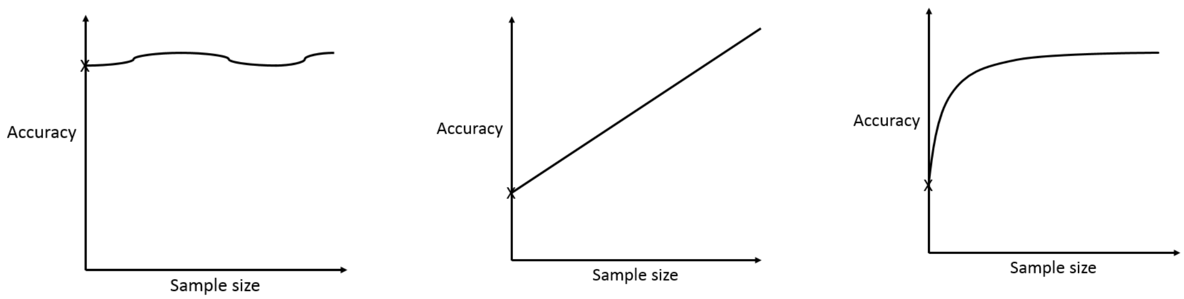
Table 5.2: Prediction results of 4 classifiers trained on *tf-idf* representation of user posts with all the features and $max\_df = 0.7$. Parameters not reported in the table were left to their default value. The target classes created based on percentage of mis-classification of a post are 'None', 'Possible', and 'Systematic'. Important features of a strong classifier can help us in identifying set of terms which are often leading to mis-classification of a post. Ignoring those features might increase the accuracy of our model.

be generalized to other MOOC courses as the terminology differs from one subject to another. (ii) As the course is an introductory course, the level of understanding or explanations differ in an advanced course. Hence this model may not perform well in an advanced course without additional training. While this chapter has demonstrated the potential of using a classification model, which can be used to automatically assess a student, many opportunities for extending the scope of this work remain. Two directions for future work are of particular interest. First, our focus was on obtaining an accurate model, which was accomplished with approximately 80% accuracy. Users may not only be interested in this overall performance, but in knowing the *confidence* of our model for each post. As text classification is a high-dimensional classification problem, there are many noise features that make no contribution to the increase of accuracy [37]. An interesting question is thus to explore how these features relate to accuracy. We explored patterns in the errors by calculating the percentage of mis-classification for each post in the data set by running our model 10 times. Based on their percentage of mis-classification, we categorized the posts into 3 classes: None (0% mis-classification), Possible (10 to 90% mis-classification), and Systematic (100% mis-classification). For example, if a post is correctly classified in all ten runs, it is labeled as 'None'; If a post is classified 6 times correctly and 4 times incorrectly, it is labeled as 'Possible'; and if a post is misclassified 10 out of 10 times it is labeled as 'Systematic'. As aforementioned, our model is highly accurate hence most posts (505 out of 803) had 0% mis-classification; the rest are shown in figure 5.4. We used different classification models (SVMs, decision trees, and random forest) to relate the features to the mis-classification class. None was able to achieve accuracy above 50% (table 5.2). That is, we could not identify which specific features were more likely to result in errors, which in turns means that we cannot currently express a level of confidence per post. Given that this confidence level may be important for the user, other methods should be explored to further explore where the errors come from. These include, but are not limited to, association rule mining [8].

Second, additional language independent features in addition to those used in the studies [2, 95, 128] can be extracted from the discussions and compared, so that more generalized classifiers can be built and employed across different courses offered in multiple languages.

Future research may also explore the use of our model on other MOOCs with comparable or larger datasets. With larger datasets, we can get rid of the randomness in the model by avoiding over-sampling of data, but still, we can under-sample the data, if there is any imbalance. The data we used to build our model, was not capturing all the classes in the IAM Framework which forced us to group some of the posts and create a class: 'other'. Given the size and nature of MOOCs, exploring the model in other MOOCs can solve the issue of insufficient data. Additionally, exploring the use our model on other MOOCs can answer a question on generalization: can this model be used in other MOOC courses?

Finally, as a way to avoid attrition, learning systems can embed a monitoring tool. This monitoring tool can be helpful to see, whether the sample size available after a certain week of the course is sufficient to build the model with good accuracy or not. In other words, it tells us how soon can instructors start monitoring their classes. The monitoring tool may support the instructors in MOOCs to identify where the students stand, during the progress of the course, when a sufficiently sized sample is available. There are at least 3 different cases regarding the amount of data needed to obtain sufficient accuracy, as summarized in Figure-5.5 shows 3 different cases:



(a) Case 1 (b) Case 2 (c) Case 3

Figure 5.5: Monitoring tool to show the sample size required to have good accuracy.

- The sample size available at the early stage of the course is sufficient to get a good accuracy (Figure-5.5-a).
- The sample size available at the beginning is not sufficient but as the course progresses the accuracy increasing linearly(Figure-5.5-b).
- The graph following a logarithmic pattern where the model can be used after certain weeks when there is no more significant increase in the accuracy(Figure-5.5-c).

CHAPTER 6

A NOVEL THREE TIER VISUALIZATION FOR CRISIS IDENTIFICATION

We explained how policy makers can interact with off-the-shelf tools such as `Jigsaw` and `IN-SPIRE` to understand public opinion from large collections of text documents. This chapter shows how Text Analytics, Visual Analytics, and Fuzzy Cognitive Map (FCM) can collectively help an analyst to understand complex scenarios such as obesity by providing an interactive visualization environment. The analyst interacts with the visualization to update, and receive feedback through a simulation model called Fuzzy Cognitive Map. The FCM shows the relationships between different factors deemed important to the analyst; this offers a valuable focus in terms of identifying key data sources and finding connected factors. Text Analytics helps the analyst keep track of important data sources amongst the vast data collected for the analysis. Our data set and the scripts we wrote to build the interactive visualization environment can be accessed at *https://osf.io/eb38c/*.

All of this chapter has been submitted as the following article:

- **Venkata Sai Pillutla**, Philippe J Giabbanelli. Iterative generation of insight from text collections through mutually reinforcing visualizations and mental models. In *IEEE Visual Analytics Science and Technology conference*.

6.1 Introduction

Domain experts have faced staggering volumes of textual information for many years [64]. Many techniques have been proposed and evaluated to support experts in generating insight from text, using Natural Language Processing [14, 103] (e.g., text summarization or topic identification) and/or interactive visualizations (e.g., via tools such as `Jigsaw` [142] or `IN-SPIRE` [157]). The type of insight that one seeks can vary widely depending on the problem. For example, policy-makers may be interested in obtaining summaries of their constituents' views regarding proposed policies [51], while marketing and communication experts may want to analyze sentiments. In a deeper analysis, experts not only want to identify the key concepts from the documents but also capture their interrelatedness. This type of analysis is particularly common in the intelligence community, where one may accumulate a lot of documents and uses them to build an explanation of a phenomenon such as political unrest [118].

Many visualization techniques can articulate relations between concepts in text documents. Word Trees [155] and latest evolutions such as `SentenTree` [75] extend the idea of Tag Clouds (focusing on displaying only the keywords) by associating keywords in the context where they occur (typically at the sentence level). However, this limits the structure that may be obtained to trees, while complex phenomena are known to often harbor loops [105]. It also doesn't convey much of the semantic relation between the words. Radial Document Visualization convey more semantic [23], but connections are limited to hyponymy (i.e., between a specific word and a broader term, such as identifying that a dog *is an* animal). Concept maps originated as means to manually organize knowledge, gaining popularity in the education community [108]. They allow for any network structure, and represent concepts as nodes (e.g., 'poverty', 'insurgency') connected by edges annotated with propositions (e.g., 'begin with', 'necessary for'). While this encodes a much richer semantic than provided by the output of either Tag Clouds or Radial Document Visualization, it comes at the expense of difficulties in automatically building a concept map from documents. Tools from the early 2000's were either fully automatic but very limited in scope (e.g., by expecting related concepts to be within the same sentence [21] or to follow specific linguistic structures [122]) or to relied on iterations of user feedback to refine the set of edges [3]. The Concept Map Miner proposed by Villalon and Calvo in 2011 provides a visualization environment where the concept map is built automatically from the text, using text mining algorithms such as Latent Semantic Analysis [151]. While this approach and related variants [148] enable the extraction of concept maps from text, these maps still have important limits given the needs of analysts. Specifically, these maps represent only a 'still' image of what is contained in the documents. As they cannot highlight the dynamics of events, they do not fully support the forecasting needs of analysts. For example, we can see what the main factors are in the text (e.g., distrust of institutions in the case of insurgency) and we can see how they relate, but we cannot assess whether the corpus obtained last week indicates a higher probability of an insurgency than in the corpus from two months ago.

Fuzzy Cognitive Maps (FCMs) can address these limitations. As they are a simulation model, they provide forecasting capabilities. In a 2013 idea paper, we suggested that analysts can go beyond concept maps by building fuzzy cognitive maps from text documents [118], but we did not cover how it may be implemented or what algorithms would be involved. The implementation of this idea has proved arduous, as might be expected from lessons learned in gradually automatizing the creation of concept maps. In this paper, we present an implementation relying on the innovative combination of text analytics, interactive visualizations, and fuzzy cognitive maps. In contrast with previous approaches that created FCMs without visualizations other than the result or input text [154, 84], we posit in this paper that interactive visualizations are the key to create high quality FCMs from text documents. Consequently, our main contribution is the development of a visual analytics environment that creates a feedback between data exploration and model refinement. The feedback loop works as follows: the analyst explores text data based on a model of the phenomenon, and refines that model based on observed patterns, which will in turn improve the exploration experience. Fuzzy Cognitive Maps provides the modeling part, text analytics suggest the refinements, and the visualization environment constitutes the key vehicle to efficiently interact with these two components.

This chapter is organized as follows. In Section 6.2, we provide background knowledge for the Visual analytics and FCM. As we rely on an innovative combination of techniques from different fields, we do not expect the reader to be familiar with all techniques and thus include a succinct contextualization of each technique within its field. The design of our environment is presented in Section 6.3, with the software implementation discussed in Section 6.4. The implementation is open source can be freely accessed on a third-party repository at `https://osf.io/eb38c/`. The overview of functionalities is also provided in the form of a video at `https://www.youtube.com/watch?v=j3Moa12UD`. In Section 6.5, we provide a concrete case study using data from the field of overweight and obesity, showing how a refined understanding of the phenomenon can be obtained through our environment. Finally, we address the technical limitations of our visualization system and provide a brief discussion on future work in section 6.6.

6.2 Background

6.2.1 Modelling component: Fuzzy Cognitive Maps

This chapter seeks to assist experts in refining forecasting models articulating the concepts and interdependencies in complex problems. Concept maps are not computational models, that is, one cannot ‘run a simulation’ on a picture of a network. However, there are several modelling techniques which resemble concept maps in the sense that they articulate relationships between factors, but also have forecasting abilities. System Dynamics (SD) is one such technique, and it can account for aspects of problems that include delays, accumulation, etc. While qualitative data such as text can be used to *inform* the development of SD models [97], the high expressive power of SD presents a major barrier to building them more automatically from text. Fuzzy Cognitive Maps (FCMs) provide more limited models, as they do not capture features such a delay or even time. Dating back to the late 80s and originating from soft computing [89], an FCM is a causal network equipped with an inference engine (Figure 6.1). It has been used in many occasion as a reasonably proxy of stakeholder 'mental model', articulating the different factors and dynamics of their interactions as understood by a stakeholder [113, 52, 50]. The fact that FCMs can be intuitively developed has made them increasingly popular in a variety of human-environmental interaction contexts [62, 30], where experts in a field of application (rather than in computer science or modelling) use FCMs to articulate their understanding of a problem given the evidence, and then test the model's predictions for a few scenarios of interest.

An FCM models the behavior of a system through three key constructs:

(i) nodes, representing concepts of the system such as states or entities. Nodes have a weight in the range [0, 1] indicating the extent to which the concept is present at a simulation step.

(ii) weighted directed edges, representing causal relationships. Their weight is from the range [-1, 1] where negative weights indicate that increases in the source node cause a *decrease* in the target node. Conversely, positive weights indicate that increases in the source node cause an *increase* in the target node.

(iii) an inference function which updates the value of each node using based on the weights of both the edges going into it and the nodes that these edges connect to. The update is applied until the values for a subset of nodes stabilize. That subset is chosen based on the application. For example, if the goal is to understand long-term trends of obesity, one would want the concept 'obesity' in a model to stabilize (Figure 6.1).

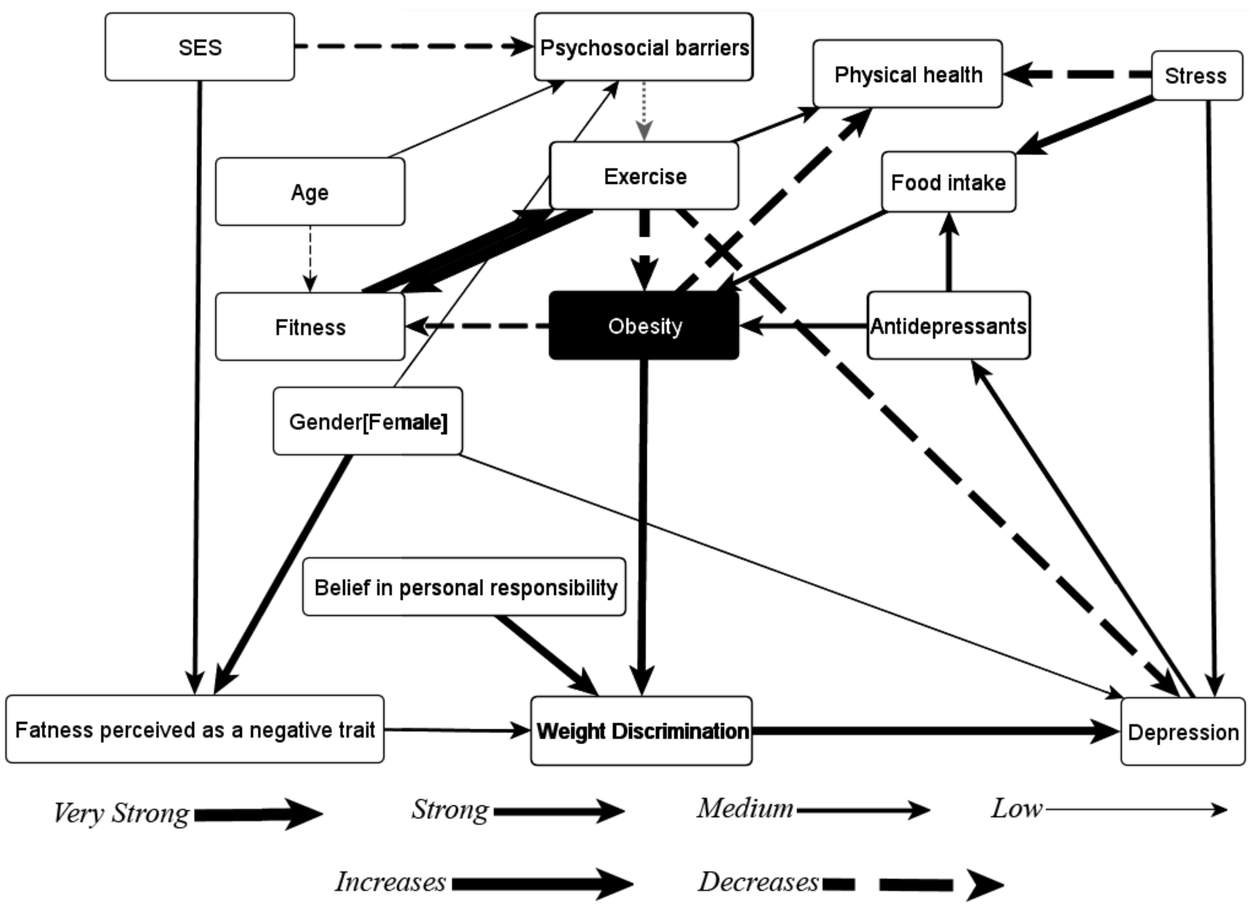


Figure 6.1: This FCM has 15 nodes/concepts (shown in boxes) and 25 interrelationships/edges. The FCM is updated until the concept 'obesity' stabilizes. Edges have either positive (solid lines) or negative (dashed lines) values, whose strength is shown as their width. The strength was obtained using Fuzzy Logic based on the knowledge of 7 experts [55].

As we seek to provide an overview of the technique rather than a complete depiction of its mathematical underpinnings, we refer the reader to [57] for a more formal introduction. One mathematical aspect will be covered, because it exemplifies why FCMs are intuitive to use, and because our proposed environment gives direct access to it. This aspect is about how experts' judgments are transformed into edge weights using *fuzzy logic*. In mathematics, a set is a collections of objects that either belong or do not, based on some definition. However, in real-world scenarios, classes of objects do not have such precisely defined criteria of membership. When an expert evaluates the strength with which a concept impacts another one, he or she may summarize it as ‘very strong’ while another may call it ‘strong’. These two linguistic variables do not map to exact values such as 4 or 5. Rather, there is a continuum of grades of membership. A ‘strong’ may be closer to a 4 than it is to a 5, while a ‘very strong’ may be closer to a 5 than a 4. Fuzzy set theory can handle this vagueness using *membership functions*, that is, a continuum of grades of membership. Experts can naturally express their judgments using linguistic variables such as ‘very strong’, which is associated with a membership function (Figure 6.2). As different experts weigh in the strength of each connection, each one of the membership functions is endorsed to some extent. These endorsements are entered into a rule based, aggregated [158, 100], and eventually de-fuzzification is applied to produce the specific number standing for the edge’s weight. In sum, fuzzy logic allows experts to use natural linguistic terms when forming evaluations, while relying on mathematics to adequately transform these terms into one number.

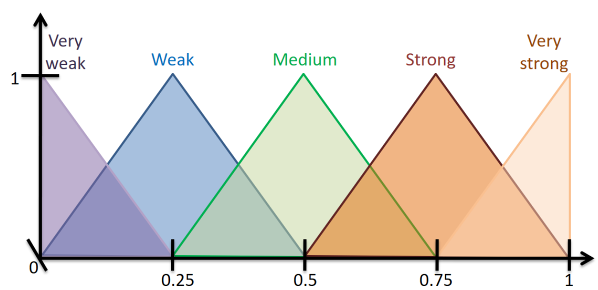


Figure 6.2: Each triangular membership function represents perception of a linguistic variable. These perceptions of linguistic terms can overlap. Overall, they define a partition of the space [86].

6.2.2 Visualization component: multi-tiered visualization

Our proposed environment for visual analytics seeks to assist analysts in navigating the text documents (through algorithms presented in the previous section) and iteravely refine their understanding of the problem (through the Fuzzy Cognitive Map presented in section 6.2.1). Several views of the data must thus co-exist in one environment, such as the text documents and the analyst's mental model. There are two broad approaches to create such environment[1]. One consists of setting up individual visualizations (e.g., one for the FCM and one for the text documents), and using brushing to work across these multiple data representations. Software such as `IN-SPIRE` [157] feature several examples of this approach. Alternatively, one can use a multi-tiered visualization, which emphasizes dependencies between the views. For example, selecting an item in one view may show how it connects to items in the other view(s) through rays, rather than using only contours or colours as would be typical in the case of brushing. Multi-tiered visualizations have been used in several works, of which three prominent are now presented using a chronological approach. These three approaches were selected as they exemplify horizontal/vertical and 2D/3D multi-tiered visualizations.

In 1996, Risch *et al.* proposed the exploratory information visualization system `Starlight` [123], which interactively generates information-dense 3D visual representations of data interrelationships. While `Starlight` possessed many interesting features, including the *linkage display system* to assist users in finding hidden patterns in the data. To show the linkages, `Starlight` uses the set of term linkages developed during the initial pre-processing stage. Clicking on an entity creates a series of rays connecting it with entities in other tiers of the visualization. For example, Figure 6.3 shows how the selection of an entity in the central tier shows its connections with items in the tier above and the tier below. `Starlight` was known to have a number of shortcomings, such as the high computational cost of text-processing algorithms to identify interrelationships, and the display complexity that came when handling a large number of entities.

In their 2008 article, Stasko *et al.* proposed a suite of interactive visualizations and named the prototype system as `Jigsaw` [142]. The main goal was to create visual representations of the information within textual documents and improve the understanding of the analysts. To achieve this, Jigsaw provides multiple views, allowing the user to visually investigate different relation-

[1]Note that when there is a direct mapping between views of the text in terms of scale, one may use multiscale text visualizations such as `Typograph` [33]. The views considered here (e.g., text documents vs. mental model) do not possess such direct mapping, thus multiscale text visualizations are not listed as one of the main options.

ships between entities in the documents where entities can be places, people, organizations and dates. While `Jigsaw` includes brushing to work across visualizations, it also has a multi-tier visualization called *list view* (Figure 6.4) which shows multiple reorderable lists of entities. For recursive relations (i.e., connecting elements of an entity list to other elements within the same entity list) such as social networks, users can play the same list side by side. Connections between the entities are shown by coloring related entities *and* drawing links between them. The brightness of highlighting of an entity represents the strength of the relationship between the entities. Compared to `Starlight`, the algorithms employed by `Jigsaw` allow it to handle document collections in the tens of thousands reasonably well. However, the display complexity can also be significant, particularly when many links appear between lists.

In 2014, Jordao *et al.* proposed the multi-tiered visualization `EduVis` to represent educational data patterns [83]. They extracted semester-wise course patterns from nine years of data from a computer science program. In `EduVis`, each layer represents one semester of a study program, displaying courses as circles.

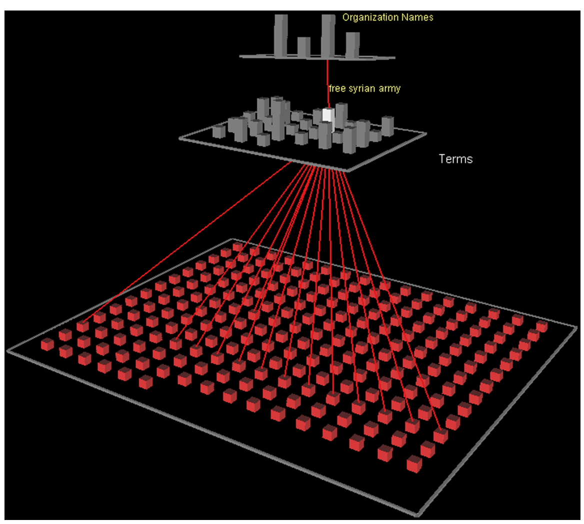


Figure 6.3: Horizontal 3D multi-tiered visualization from the `Starlight` software as exemplified in [118].

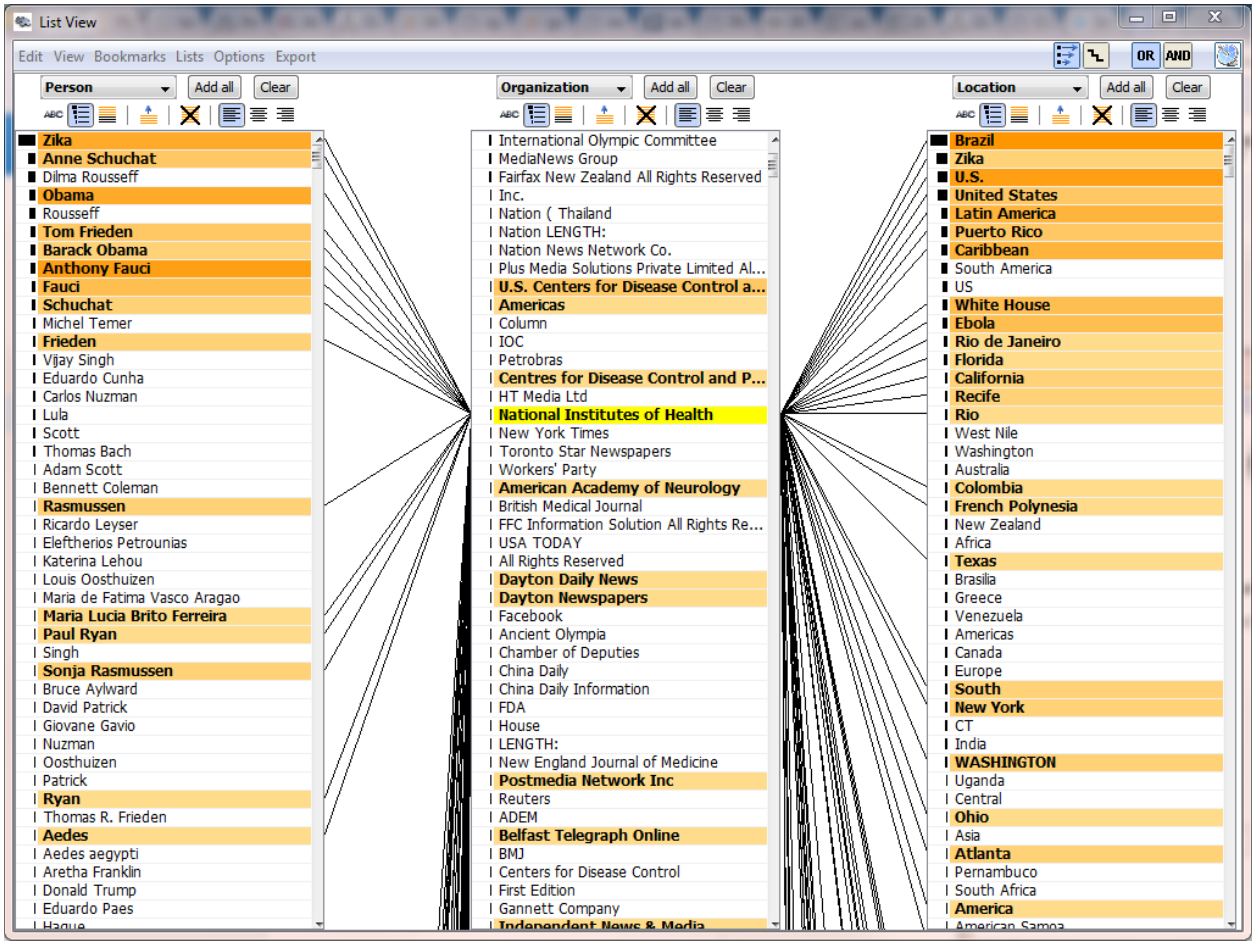


Figure 6.4: Vertical 2D multi-tiered visualization from `Jigsaw` [142].

When data on failure is available, the circles are subdivided into two semi-circles. In Figure 6.5, the green semi-circle on the left depicts the number of successful students, and the red semi-circle on the right depicts the students who failed the course. This allows users to contrast success rates for specific courses. The visualization also allows users to see the relationships of the course with others by moving the mouse over the circle. The relationships are represented by visual connectors (cubic Bezier curves) whose thickness corresponding to the number of students involved. The color of the visual connector represents success (from blue to green) or failure ranges (from yellow to red).

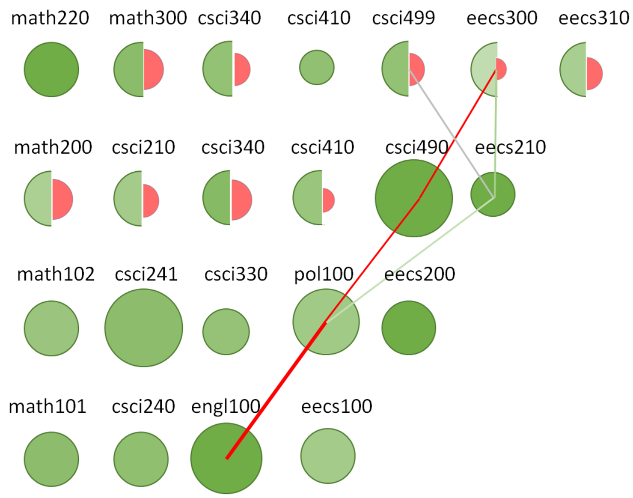


Figure 6.5: Horizontal 2D multi-tiered visualization exemplifying the principles of `Eduvis` [83].

6.3 Design of the visualization environment

6.3.1 Overview

In our 2013 idea paper, we suggested a design in 5 levels (Figure 6.6a). It proposed to sub-divide the FCM into the top two layers, and included a data source layer at the bottom. Based on feedback from our idea paper and ensuing discussions on the design, sub-dividing the FCM did not appear pertinent because (i) analysts would have to decide what is a micro- versus macro-factor, and (ii) this decision is not useful since analysts can just focus on the factors whose long-term trends are of interest. Therefore, our current design includes the whole FCM as the top layer. We also removed the data source layer as many of the datasets we worked with did not include that information. For example, intelligence gathering sessions would only provide the transcript of interviews but not who the source was, or where it took place. In sum, our environment for visual analytics is composed of three tiers (Figure 6.6b).

The overall visual encoding of information and interaction builds on lessons learnt from previous multi-tier visualizations (section 6.2.2). As in `Jigsaw`, `STARLIGHT` and `Eduvis`, connections between elements in the three tiers are shown through rays. As in `Jigsaw`, we further emphasize these connections through colors. In our case, the contour of elements indicates that they are related. Similarly to `STARLIGHT` we use a horizontal 3D approach, which was found to be in-line with other software that the intelligence community is familiar with [118]. As in `STARLIGHT`, despite the fact that the rendering is performed in 3D, we limit the user's control of the viewpoint to avoid disorientation effects. Since previous papers highlighted that the display complexity increased notably with the number of connections found across tiers [142, 123], we considered it essential to support the use of our environment on large displays (which can unclutter the rendering), and particularly large touch screens. To ensure that interactions with such apparatus could be intuitive, there are no double or right clicks: a single (left) selection suffices for all operations. In addition, the same selection always results in the same effect: users do not need to change between tools.

Similarly to previous IEEE VIZ papers having multiple parts [67, 16], the remainder of this section explains each of the (three) parts and then details how analysts interact with them.

6.3.2 Design of the three layers

Fuzzy Cognitive Map: concepts and weighted causal edges

The first module or the top layer is the Fuzzy Cognitive Map. The nodes of the FCM correspond to abstract concepts (e.g., coordination, violence, negotiations, military action) and the edges of the FCM correspond to the strength of causation from one factor to another. The weights of the concepts are updated based on the strength of causation from one factor to another which can be obtained from the text files with minimum intervention from the analyst. At any point of the analysis, the analyst can hover over any FCM Concept to see its name and weight.

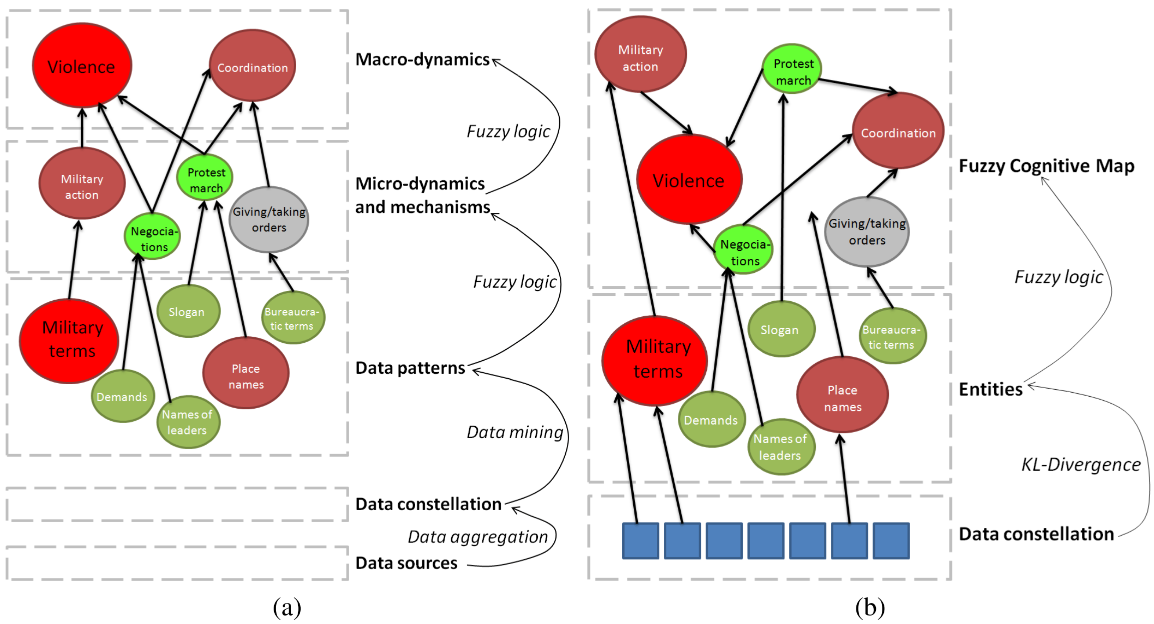


Figure 6.6: (a) Design as suggested in our previous idea paper [118]. Note that the type of data 'patterns' was unknown, and the critical linkage between these patterns and the data constellation was not specified beyond the fact that data had to be mined. (b) New design proposed and implemented in this paper. The FCM is entirely within the top layer, and does not position factors as micro- or macro-. The data constellation is made of individual text documents, linked to entities using KL-divergence.

Entities: list of terms

The entities are the key terms, and are represented as a collection of colored rectangles in the second or middle layer. The coloring criteria for the second and third modules will be explained in the subsequent paragraphs after explaining the third module.

Data constellation: set of text documents

The Third module or the base layer is a collection of text files which can be loaded by the analyst. The *tf-idf* scores are computed to extract important keywords from the text and help the analyst in identifying new entities. Terms with either very high or very low *tf-idf* scores were considered noise and are not shown as suggestions. The analyst can click on a file in the base layer to see these suggestions from that file in a separate rectangular box (Fig 6.9).

6.3.3 Interactive Analysis with the Visualization

The analyst can click on a document to see a list of suggestions; these are a set of key terms extracted from the file based on their *tf-idf* scores. The analyst can add potential entities from the suggestion box into the entity layer by clicking the add icon which is next to every suggestion. The analyst can also delete an inappropriate entity from the middle layer by clicking on the 'Remove Entity' button. This action pops up a new frame asking for the name of the entity to be deleted.

If the analyst wishes to add new concepts to the FCM, which can be achieved by clicking the 'Add Concept' button. This action will pop up a new window asking the analyst to input the concept's name and weight. Similarly, the analyst can add or delete FCM relationships by clicking on 'Remove Link' or 'Add Link' buttons respectively.

The relevance of each document to a particular entity is computed using KL-divergence (explained in section 2.5.2). Once, all the modules are loaded, the entities in the middle layer are colored based on their score. The scores of the entities are computed as follows. Each entity is treated as a query term and the documents which are having a KL-divergence score greater than zero for the entity are connected to the entity. For example, if there are 2 documents: a and b in the base layer; The KL divergence score from the documents to the entity, 'diabetes' is 0 and 0.45 respectively; then only document b is connected to the entity. One disadvantage of using KL divergence as a ranking function is that scores are not comparable across queries. To compare the scores across all the documents, the individual KL-divergence scores of the documents for each entity are normalized using min-max normalization. Min-max normalization brings the value to the range of 0 and 1. The sum of the normalized KL-divergence scores of all the connected files of an entity are summed to generate the score of the entity, which is intuitively the amount of relevant information contained in the documents for a particular term. The color of the entity is used to show this score (green representing the lowest and red representing the highest). Similar to the FCM module, the analyst can hover over the entity or the file to see the name and the KL-divergence score.

Once all the three modules are loaded into the system, the analyst can perform interactive exploration. The analyst can check the relationships between elements in different layers by clicking on any individual component. For example, clicking on an FCM concept will show the relationships to the connected entities and the list of files connected to those entities. These connections are represented using white lines between the components. Similarly, clicking on an entity will reveal all the connections from the selected entity to the files in the base layer and the FCM concepts. In the same way, the analyst can observe the relationships from a selected file to the list of connected entities and the list of concepts connected to these entities (Fig 6.9).

Furthermore, the analyst can weight the relationship between an entity and an FCM concept, if one exists between them, by clicking on the link or create a new relationship by clicking 'Add Link' button. The entity is treated as a virtual concept of the FCM; link between the entity and the connected FCM concept is treated as an edge. The analyst is prompted to provide (i) the direction of causality (i.e., whether the target increases or decreases as a function of the source), and (ii) its strength using linguistic terms (Figure 6.9) which will be converted into a crisp value using fuzzy logic (Section 6.2.1). The FCM starts evolving with its new virtual concept (entity) until it stabilizes (Figure 6.10). After the FCM evolution, if the analyst wishes to undo the simulation back by one step, he can click the undo button provided in the tool.

6.4 Implementation

We have implemented the visualization system using `Processing 2.2.1` framework. The algorithms contained in the visualization system are implemented in `Java 1.8`. We used `jung 2.1.1` library for the graph data structures to store the FCM concepts and relationships. The FCM implementation is derived from an implementation used in several previous works including [54, 50, 52].

When the analyst loads the files into the system, each file is pre-processed sequentially. We used `Stanford CoreNLP 3.4` to pre-process the text. The list of suggestions are generated for each text file after performing tokenization, stop word removal, and lemmatization operations (which are common across all document retrieval techniques). These suggestions are then sorted based on their tf-idf scores and only the top-10 suggestions in the list are shown to the user. Processing these documents and generating the tf-idf scores and determining the relationships between the files and the entities may take several minutes on a personal computer as these steps were implemented sequentially.

There are certain parameters in our implementation. We determined the connections between the entities and the files using KL divergence. A connection between the entity and the file is established if the KL divergence score is greater than zero. However, the actual KL divergence score can be seen by hovering over entity. We tuned the FCM stabilization constant, which is the stopping criterion for the FCM. We set this value as 0.005. To ensure that the value of each concept in the FCM remains in the interval [0, 1], we apply the $tanh$ function and limited the output to the [0, 1] interval. We combined the expert opinions for each relationship using the Mamdani algorithm [100], the sum method of aggregation, and the centroid method for defuzzification.

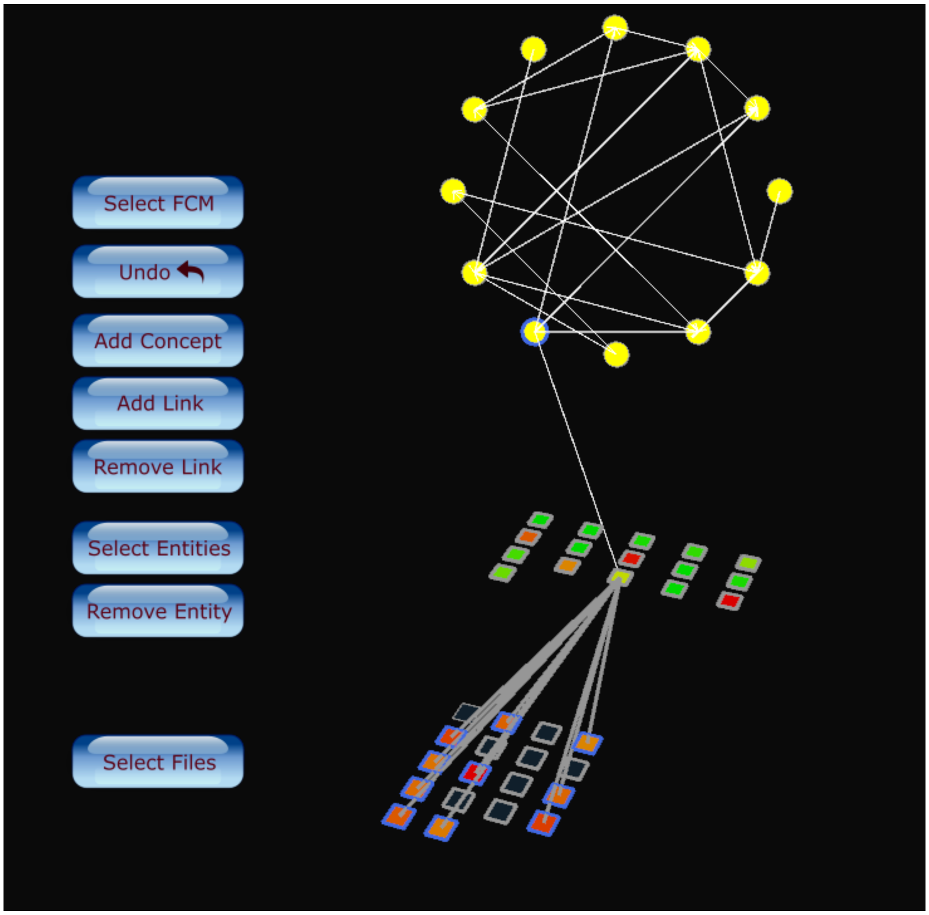


Figure 6.7: Analyst can check the relationships between an entity, its connected concepts in the FCM, and the files by clicking it.

6.5 Case study: overweight and obesity

In this section we will discuss a case study showing the usability of our visualization system to understand complex scenarios. The Mind, Exercise, Nutrition, Do it (MEND) program is an intervention for children with obesity and their families. The program consists of sessions on nutrition and behavior change, as well as group physical activity [131, 53]. While MEND originates from the UK, it has also been implemented in Australia, Canada, Denmark, New Zealand and the US. The program is designed so that it can be implemented in the community by people who are

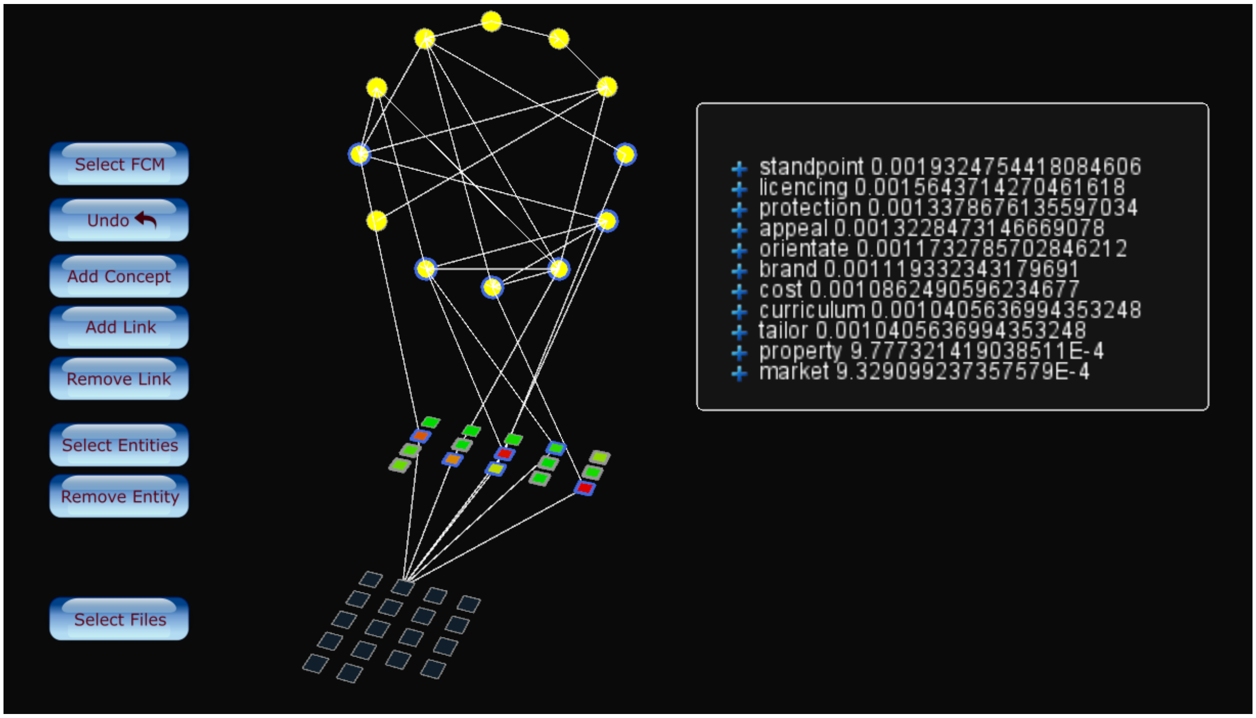


Figure 6.8: Relationships between a selected file, its connected entities in the second layer, and their connected FCM concepts. The suggestion box on the right side shows a list of terms with good *tf-idf* scores from the selected file.

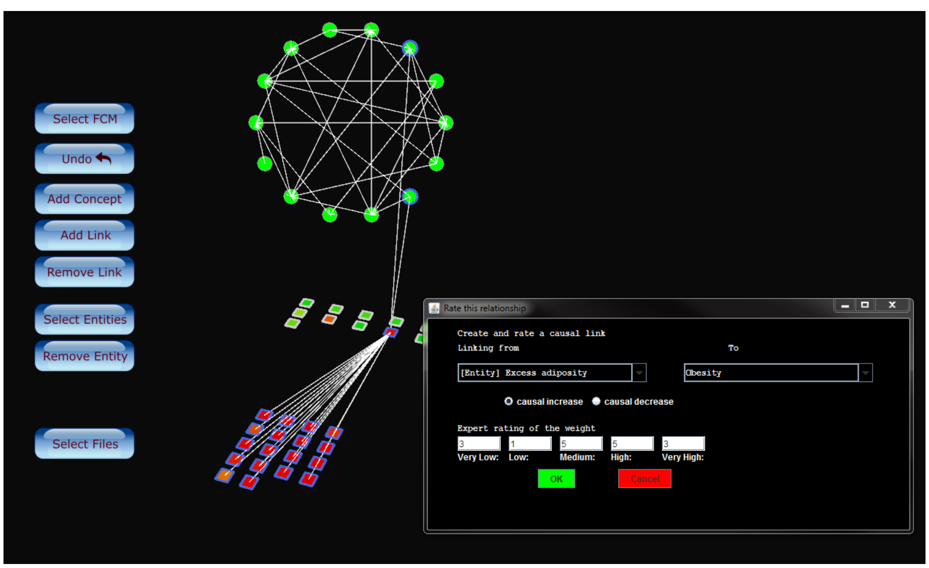


Figure 6.9: Weighting the relationship between a selected entity and one of its connected FCM concepts.

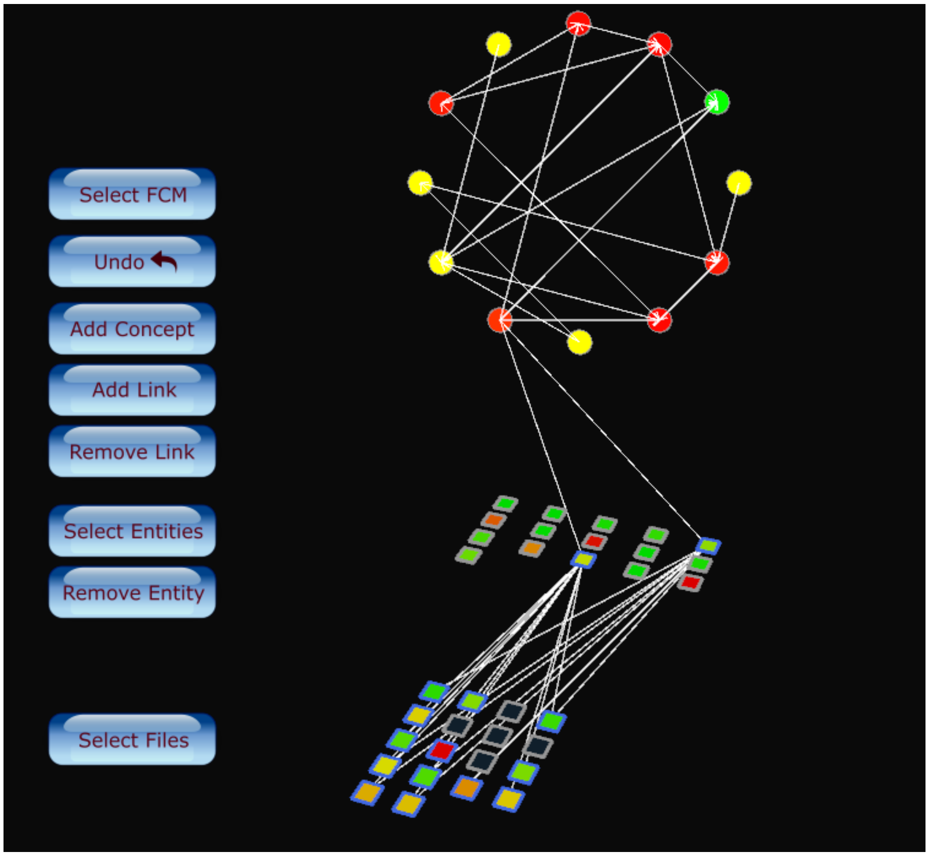


Figure 6.10: FCM after evolution.

not health professionals. Due to the scaling-up of the program, both in its geographic reach as well as the number of participants, it is of particular interest to understand the complexity of the organizational structure.

In order to capture the vision of the stakeholders, interviews were performed and transcribed by the Chronic Disease Systems Modeling laboratory at Simon Fraser University, which provided ethics approval. A total of 18 stakeholder interviews were produced by the Chronic Disease Systems Modeling laboratory. In this case study, the analyst created an FCM to understand the behavior of the system and supported it with the help of the vision expressed by the stakeholders.

The interview transcript files are first loaded into the visualization system. After getting a basic idea of the key terms by going through all the suggestions, the analyst clicked on each of the files and added the obesity related key terms from the suggestion box into the entity layer as discussed in section 6.3.3. The key terms added by the analyst are ‘waist’,‘circumference’, ‘sweetener’, ‘healthy’, ‘treatment’, ‘parenting’, ‘diabetes’, ‘clinical’, ‘guide line’, and ‘curriculum’(Figure 6.11 (a)). Terms with higher tf-idf scores such as ‘standpoint’, ‘licensing’, ‘technology’, etc. are left behind as they are not related to the obesity epidemic, which is the main focus of the analyst.

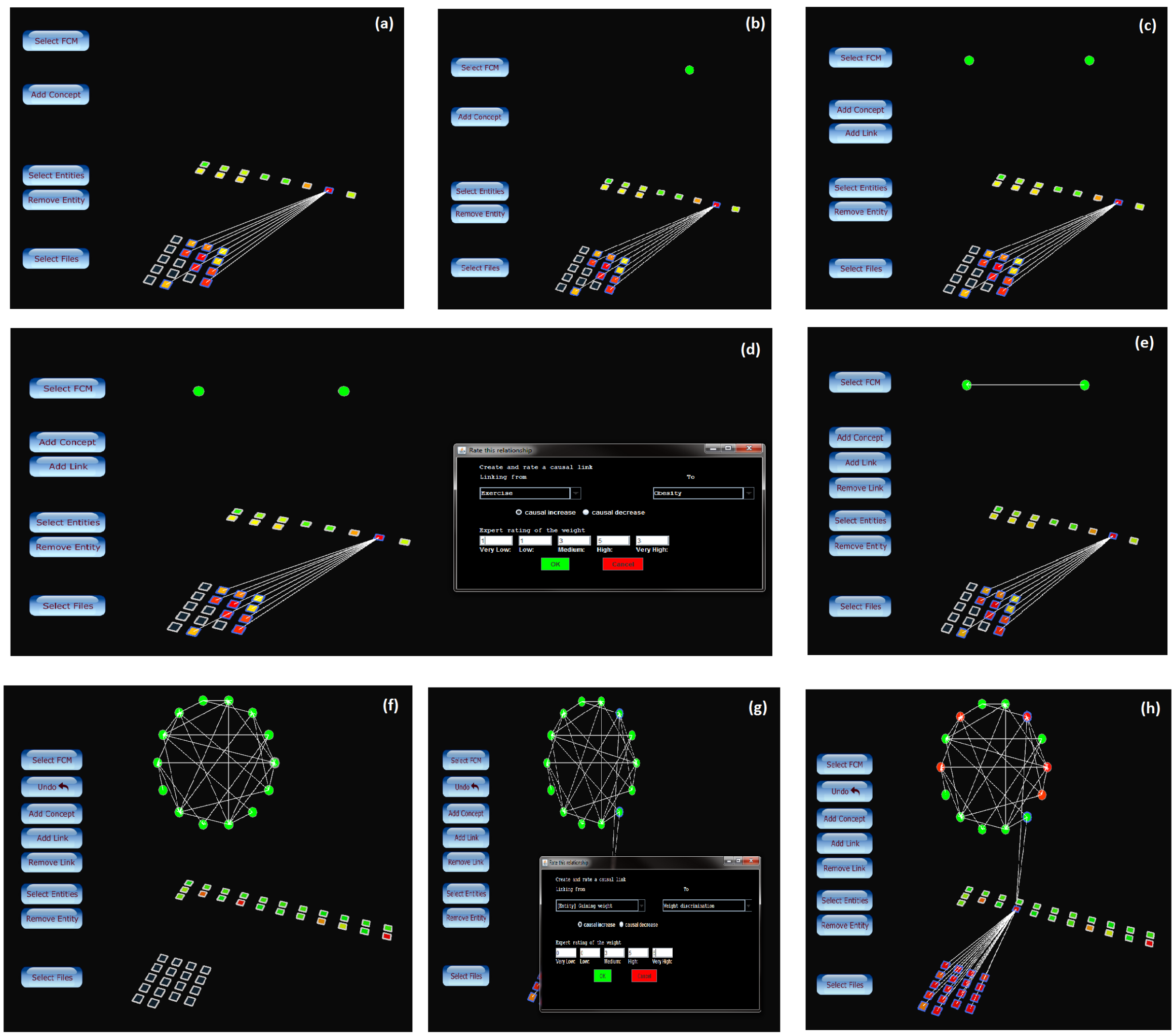


Figure 6.11: Case study showing the FCM creation from text files and refining the FCM to understand the causal relationships of Obesity epidemic.

To identify the impact of these entities on the overall system, the analyst started building the FCM by adding concepts. He added a concept, ‘obesity’, which is a super-set of the entities ‘waist’,‘circumference’, and ‘diabetes’ (Figure 6.11 (b)). He then added another concept, ‘exercise’ to see the impact of exercise on obesity (Figure 6.11 (c)). He created a relationship between the concepts by clicking the ‘add link’ button. He also created links between the concepts and the entities by clicking the ‘add link’ button and providing the details of the relationship from the group of experts. Thus, he created a full chain from the files to the concepts in the FCM.

The analyst observed that the values of the FCM concepts have changed after the evolution. But however, he noticed that the there are few more entities which are related to the epidemic. He created a text file with all those entities and uploaded the text file into the system. By observing the KL divergence scores, the analyst identified that some of the newly added entities are strongly supported by the interview files. To understand the impact of these newly added factors on the key concept, obesity, the analyst added new concepts which are supersets of the newly added entities (Figure 6.11 (f)). He identified entities with high scores such as ‘health issues’ and ‘gaining weight’ and connected them to the FCM concepts. After the FCM evolution, he noticed that the concepts, ‘Food Intake’ and ‘Anti Depressants’, are strongly impacting Obesity (Figure 6.11 (h)).

This kind of analysis provides two immediate benefits in the process of understanding complex scenarios. First, it helps in identifying concepts and relationships that appropriately cover the experience of stakeholders. Second, it can help in creating new relationships between concepts and entities and see their impact on the overall system.

6.6 Discussion

We developed a novel visualization system which is a synergy of visual and computational methods. By leveraging the strengths of text analytics, visual analytics, and Fuzzy Cognitive

Maps our visualization system is able to help modelers and analysts understand complex systems. The analyst can either follow a top-down approach or a bottom-up approach to explore complex phenomenon. In other words, the analyst can load the files alone and start identifying entities, concepts and then build the relationships. The analyst can also load the FCM first and then load the entities and text files to see the relevance between the entities and text files and then refine the FCM based on these insights. The implemented system has at least two major advantages: (i) the analyst can interactively build an FCM from scratch and support it with evidence from the text files; (ii) the analyst can explore large text files and identify important terms and also quantitatively explain their impact on the system.

One shortcoming of our visualization is the increase in display complexity when operating on a large collections of entities or text files. Experimenting fisheye views [68] to selectively focus and reveal details as user approaches a particular region of visualization may be helpful in reducing the display complexity. Further usability studies are needed to empirically validate our system and understand how actual users interactively explore scenarios using this system.

One disadvantage of using KL divergence as a ranking function is that scores are not comparable across queries. This does not matter for adhoc retrieval, but is important in other applications such as topic tracking. We compared the average KL divergence scores of the entities by normalizing them using min-max normalization. Kraaij and Spitters [90] suggest an alternative proposal which models similarity as a normalized log-likelihood ratio (or, equivalently, as a difference between cross-entropies). Future work may thus experimentally determine the efficiency of different methods to normalize the KL divergence, including the min-max method used here and the log-likelihood ratio proposed by Kraaij and Spitters.

we see two principal directions for future work. (i) experimenting fish eye views to reduce the display complexity. (ii) exploring alternative layouts for graphs to avoid overlapping edges in the FCM.

We briefly discussed about graph layouts in section 2.2. Graph layout algorithms are used to visualize graphs and enhance their legibility [144]. Researchers focus on aesthetic criteria such as minimizing edge crossings. Dickerson et al. [28] used dot layout to visualize metabolic and regulatory network models. As a part of future work, other alternative layouts such as radial layout, hierarchical layout, etc. may be explored to enhance the visualization.

CHAPTER 7

CONCLUSION

The overarching goal of this thesis was to develop systems for social computing, both in the context of online communities and the study of societal processes. Specifically, these systems should allow users to learn while in turn learning from the users (positive feedback). This goal was subdivided into two specific aims, thus assessing the benefit of this feedback in both aspects of social computing:

1. Can instructors in Massive Open Online Courses (MOOCs) use an automated system to refine their assessment of the students' learning experience, while refining the system's own assessment?

2. Can a system guide analysts and modelers in exploring data about social processes, while learning from the users how to best guide them?

To answer (1), we built a machine learning model to automatically categorize students' discussions into appropriate phases of the IAM Framework. Our results show that text data, which is high dimensional, was linearly separable. It can be understood from the results in table 5.1 that SVC Linear is giving a better accuracy than the other classifiers. However, accuracy may be improved by getting more training data which we are in lacking in our case. We successfully created a model support assessment of the students but, we weren't able to refine the system's own assessment because there wasn't any pattern in the mis-classification of interactions. Considering a different MOOC course data set might be helpful in refining the system's own assessment.

Our classification model can support many MOOC stakeholders in automatic assessment of students, which is costly to evaluate manually. The model can be used in performing large scale analysis of students' performance based on their discussions in MOOC forum, and answer questions such as, (i) are students answering fellow students' questions performing better than others? (ii) Do students who express gap in their understanding, drop out eventually?

There are some limitations of our work: (i) Since the data set is from a chemistry MOOC course and the classification model built on the uni-grams of chemistry terminology, it cannot be generalized to other MOOC courses as the terminology differs from one subject to another. (ii) As the course is an introductory course, the level of understanding or explanations differ in an advanced course. Hence this model may not perform well in an advanced course without additional training.

To answer (2), we created a visualization system for the analysts to interactively explore data about social processes by interacting with the visualization to update, and receive feedback through a Fuzzy Cognitive Map. we emphasized how text analytics, visual analytics, and Fuzzy Cognitive Mapping can be combined to aid the analysts in gaining new and deep insights into complex real world scenarios.

Many directions for future work have been suggested, for example, switching to a 2-dimensional view may help in reducing the visual complexity of our system. Besides visual complexity, we have used *Processing* framework to build the visualization, As *Processing* is continuous rendering framework, the memory load was high for exploratory tools, exploring other options to render the visualization is another point of interest for future work.

In the case of classification of student discussions, one important problem we faced was imbalance of the dataset. We solved this problem by oversampling the existing data using Synthetic Minority Over Sampling Technique(SMOTE). However, the data set size was also very small to build the machine learning model with higher accuracy. With larger samples, we can also avoid randomness caused by oversampling of the data.

Furthermore, the data we used to build our model, was not capturing all the classes in the IAM Framework which forced us to group some of the posts and create a class: 'other'. Given the size and nature of MOOCs, exploring the model in other MOOCs can solve the issue of insufficient data. Additionally, exploring the use our model on other MOOCs can answer a question on generalization: can this model be used in other MOOC courses?

Another important point to work on in the future is building a monitoring tool to tell the instructors how soon they can start monitoring their class. As mentioned in the chapter 5, a monitoring tool may be created to check whether the sample size obtained in the course after a particular week is sufficient to build the model with good accuracy.

In the next section, we will present some ideas to extend this work towards my PhD study.

7.1 Future research goals

Word-sense disambiguation is an open problem of Natural Language Processing. When a word in a sentence has multiple meanings, word-sense disambiguation is the process of identifying the right sense or meaning of the word. In section 2.4.3, we explained the usage of lexical chains for word sense disambiguation. I would like to further explore how artificial neural networks, more specifically recurrent neural networks, can identify the true sense of a word given the part of the sentence before the ambiguous word.

We have explained query based document retrieval techniques in section 2.5.2. I am also interested in identifying alternatives to traditional document retrieval techniques. COSIMIR (Cognitive Similarity learning in Information Retrieval) [101] is a feed forward neural network of back-propagation type to determine the query-document relevance. I would like to investigate the usage of feed forward and self organizing neural networks for this task.

APPENDIX

CONVERSATIONAL SYSTEMS

A Chatbot is a conversational agent that interacts with users in a certain domain with natural language sentences [76]. Chatbots can be used in a variety of settings that include Online Discussion Forums [76] and Tutoring systems [40]. There are two main ways to design conversational systems: either the system asks questions (e.g., tutoring systems), or the user asks questions (e.g., online flight booking systems, IT helpdesk troubleshooting [152]). As shown in figure A.1, these can be further classified into 2 types: rule based chatbot in which the rules are manually authored and a machine learning based chatbot in which the conversation is based on the training set of the model.

This Appendix will first provide an overview of selected chatbots that have hardcoded rules for the system to ask questions. Then, we will provide an example of a system in which the user asks questions, but the discussion is still not based on learning. Finally, we will point to recent developments in the field with the use of machine learning, and deep neural networks in particular. Figure A.1 shows how we classified conversational systems in this Appendix.

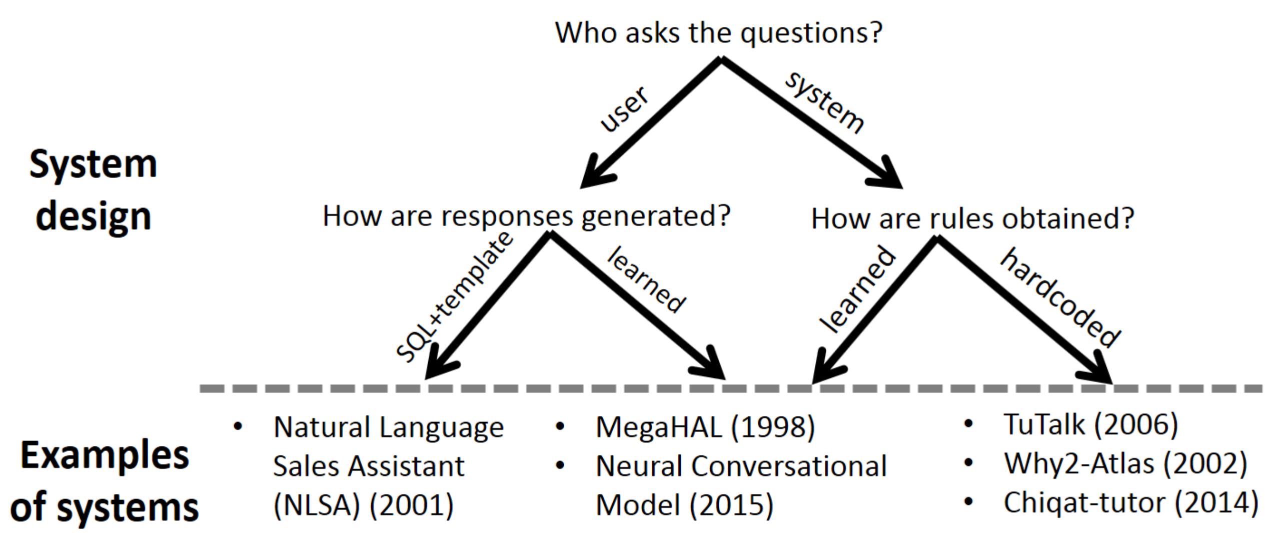


Figure A.1: Classification of conversational systems

Conversational systems either have a modular architecture with hardcoded rules or have a machine learning model that can ask questions or answer users' questions based on the provided training. *TuTalk*, *Why2-Atlas*, *NLSA*, and *ChiQat-Tutor* (2014) have a modular architecture. In other words, they have individual modules which have specific goals and are wired together. Both *TuTalk* and *Why2-Atlas* tutoring systems involve knowledge construction dialogues(KCDs), a main line of reasoning that tutor tries to draw out from the student through a series of questions [82]. Tutalk, Why2-Atlas and ChiQat-Tutor have dialogue managers which are a network of finite number of states. State change is initiated based on triggering event or condition. This type of network is referred to as Finite state-(or graph-) based systems. Alternatively, there are systems which are implemented as frame based systems (eg. NLSA) in which system fills slots in pre-defined templates to answer users' questions [104].

Jordan *et al.* proposed a dialogue system, TuTalk (an acronym for **Tu**torial **Talk**) to support the rapid development of dialogue systems in learning studies. TuTalk is a collection of core dialogue system modules and resources that together form a system that can handle users input, understand it, generate the output, and handle the output along with the discourse, an end-to-end tutoring system. TuTalk performs these actions using modules such as Coordinator, Input handler, output handler, Understanding, Generation, Student module, Dialogue manager, and dialogue History manager. The details about each of these modules were clearly explained in [82].

TuTalk's dialogue manager is a finite-state network with a stack and is implemented using the reactive planner APE [41]. The flow of the dialogue is specified as a set of states with transitions denoting alternative paths. A state in the finite state network is either a push to a sub-network as with the right-most and left-most nodes in Figure A.2 or an initiation that expresses some concept plus additional response as with the top node and its three branches in Figure A.2. There is a sub-network for each complex topic to discuss in dialogue. The concept expressed by a response is part of the dialogue context that can help decide the next state to which to move. Few other conditions that determine which state to go next were explained in [82].

The author writes a recipe to achieve a goal. When writing a recipe for a topic, the author can create one or more steps which can also be paired with a set of anticipated responses (first step

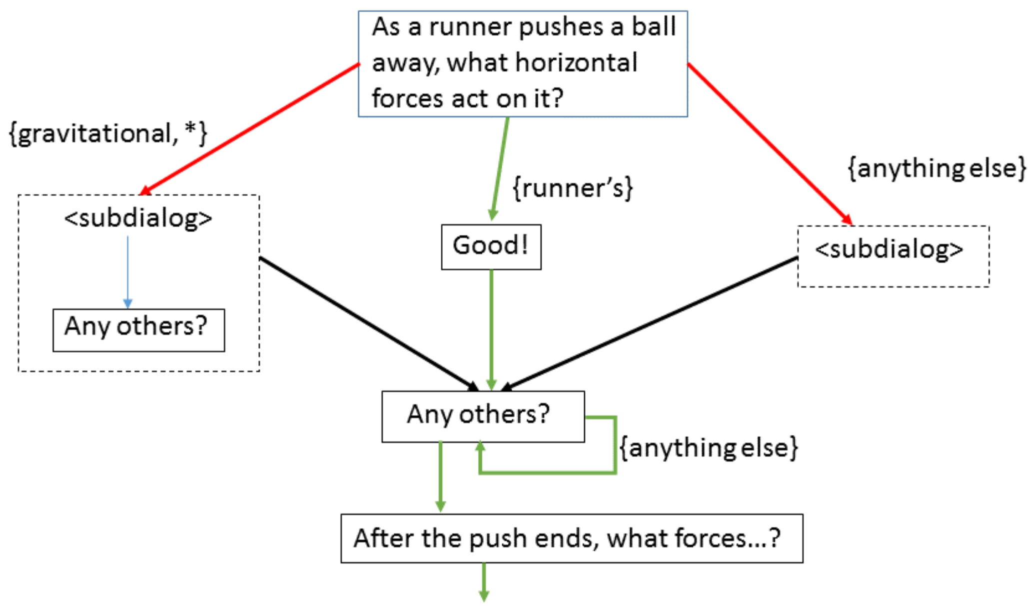


Figure A.2: Finite State Network with response concepts and optional steps

in figure A.3). A recipe can embed another recipe by referring to the goal name of that recipe (last step in figure A.3). Responses can be authored to include follow-up actions so that flawed or vague responses can be addressed as needed.

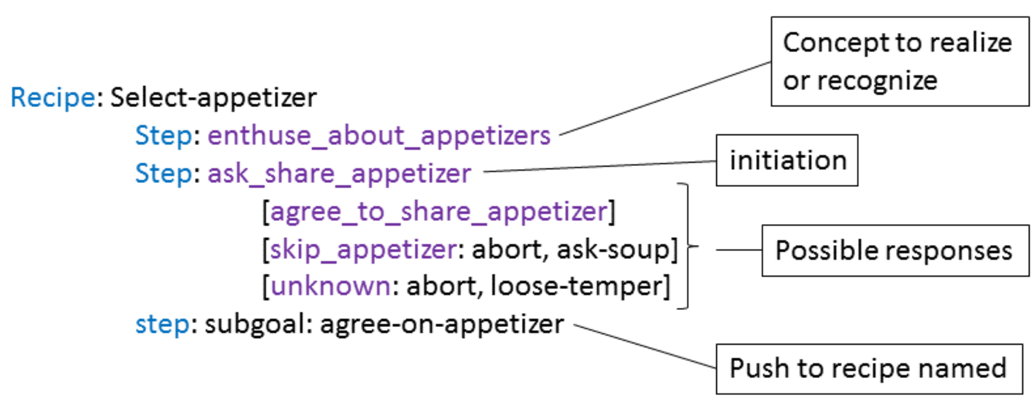


Figure A.3: TuTalk basic script showing recipes and steps

VanLehn *et.al.* proposed a dialogue based tutor for Qualitative physics essay writing, Why2-Atlas [149]. This system teaches qualitative physics by having students write paragraph-long explanations of simple mechanical phenomena.

Why2-Atlas system consists of many modules: A sentence-level understander (SLU), a discourse-level understander (DLU), a tutorial strategist, and a dialogue engine. All these modules are controlled by a discourse manager. The SLU converts each sentence in the student's essay into a set of prepositions. The prepositions can be expressed in first-order logic. The functionality of each module is explained in [149].Ungrammatical inputs are handled by the parser by skipping words, inserting missing categories and relaxing grammatical constraints as necessary in order to parse the sentence.

AlZoubi *et.al.* proposed a modular tutoring system, ChiQat-Tutor [4] which focuses on tutoring computer science data structures(e.g., stacks, trees) and algorithmic strategies(e.g., recursion). ChiQat-Tutor uses Recursion Graphs (RGraphs) which are a visual representation of recursive execution[1]. The recursion module in ChiQat supports five individual tasks [4]: (1) tracing an RGraph; (2) validating an RGraph; (3) constructing an RGraph; (4) animating an RGraph; (5) answering multiple choice questions.

The space of possible inputs and outputs in the tutoring systems is limited. As they try to achieve a very specific goal of teaching a few concepts, the users need not drive the conversation. But, there are systems for few applications which are based on the users' questions such as a travel system which has the goal of helping a traveller find an appropriate train or flight.

Chai *et.al.* proposed a 'Natural Language Sales Assistant (NLSA)' that helps users find relevant information about products in e-commerce sites. NLSA system uses a shallow natural language parser for noun phrases to extract information of interest from the user query and generates a well formed XML [19]. This system has a frame based dialog manager which organizes the results

[1]A graph $G = (V, E)$ is recursive if $V \subseteq N$ and $E \subseteq [N]^2$ are recursive [45]

from the knowledge base into a template. NLSA‘s dialogue manager uses conversational discourse history when generating responses to the user.

Hutchens and Alder were the first to propose a machine learning based chatbot, MegaHAL [77] to generate replies from a training corpus. The chatbot's language model consists of two Markov models to predict the words that *precede* (Backward Model) or *follow* (Forward Model) a sentence, respectively. MegaHAL extracts the keywords from the user input after removing the stop words. Which reply should be the output to the user is decided using a formula that defines the *highest information*. However most of the sentences generated were un-grammatical. MegaHAL suggested that it was possible to learn from users instead of pre-imposing all the rules, although technical improvements had to be achieved to generate well-formed sentences. Such improvements have been brought forward using machine learning techniques, such as Deep Neural Networks (DNNs). DNNs are powerful models that achieved excellent performance in existing complex learning tasks such as speech recognition [73].

A deep neural network is a artificial neural network that has more than one layer of hidden units between its inputs and its outputs [73]. DNN's are trained by back-propagating derivatives of a cost function that measures the variation between the actual outputs and the target outputs [130]. Using deep learning for NLP applications has been studied by several people [9, 71, 24]. Recurrent neural networks(RNN) are one of the most popular architectures used in NLP problems as their architecture is suitable to process variable-length text [139].

Vinyals and Le proposed a 'neural conversational model' using a specific recurrent neural network architecture, Long Short-Term Memory (LSTM), which reads the input sequence token by token, and predicts the output sequence, one token at a time [152]:

> "Long Short-Term Memory (LSTM) is a specific recurrent neural network (RNN) architecture that was designed to model temporal sequences and their long-range dependencies more accurately than conventional RNNs [132]."

The output sequence is given to the model during training, so that it learns using back propagation. The model converses by predicting the next sentence given the previous sentence or sentences in a conversation. This method is based on a ‘seq2seq framework’ [145] which uses a specific recurrent neural network architecture with memory blocks in hidden cells, multi-layered LSTM to map the input sequence to a fixed dimensionality vector and then another deep LSTM is used to decode the target sequence from the vector. This model answers users’ questions based on the training set, and hence it requires much fewer hand-crafted rules. It can be used across different domains by training it with different corpora.

BIBLIOGRAPHY

[1] C. C. Aggarwal and C. Zhai. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer, 2012.

[2] A. Agrawal, J. Venkatraman, S. Leonard, and A. Paepcke. YouEDU: Addressing confusion in MOOC discussion forums by recommending instructional video clips. In *National Science Foundation*. Stanford InfoLab, February 2015.

[3] A. Alves, F. Pereira, and A. Cardoso. Automatic reading and learning from text. In *Proceedings of the International Symposium on Artificial Intelligence*, pages 302–310, 2001.

[4] O. AlZoubi, D. Fossati, B. Di Eugenio, and N. Green. ChiQat-Tutor: An integrated environment for learning recursion. In *Proceedings of the Second Workshop on AI-supported Education for Computer Science (AIEDCS). Honolulu, HI*, 2014.

[5] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernal functions. *Neural Networks*, 12(6):783–789, July 1999.

[6] R. An. Effectiveness of subsidies in promoting healthy food purchases and consumption: A review of field experiments. *Public health nutrition*, 16(7):1215, 2013.

[7] R. Barzilay and M. Elhadad. Using lexical chains for text summarization. *Advances in automatic text summarization*, pages 111–121, 1999.

[8] E. Belyi, P. J. Giabbanelli, I. Patel, N. H. Balabhadrapathruni, A. B. Abdallah, W. Hameed, and V. K. Mago. Combining association rule mining and network analysis for pharmacosurveillance. *Journal of Supercomputing*, 72(5):2014–2034, 2016.

[9] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[10] M. W. Berry, Z. Drmac, and E. R. Jessup. Matrices, vector spaces, and information retrieval. *Society for Industrial and Applied Mathematics Review*, 41(2):335–362, 1999.

[11] A. Beykikhoshk, O. Arandjelović, D. Phung, S. Venkatesh, and T. Caelli. Using Twitter to learn about the autism community. *Social Network Analysis and Mining*, 5(1):1–17, 2015.

[12] K. Bielenberg. *Groups in social software: Utilizing tagging to integrate individual contexts for social navigation*. PhD thesis, Universität Bremen, 2005. pages 71-97.

[13] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*, pages 221–228. O'Reilly, 2009.

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Advances in neural information processing systems*, 1:601–608, 2002.

[15] T. Bodnar and M. Salathé. Validating models for disease detection using Twitter. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 699–702. ACM, 2013.

[16] H. Bosch, D. Thom, F. Heimerl, E. Püttmann, S. Koch, R. Krüger, M. Wörner, and T. Ertl. Scatterblogs2: Real-time monitoring of microblog messages through user-guided filtering. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2022–2031, 2013.

[17] T. Brandes, UlrikErlebach. *Network analysis*, pages 131,295. Springer, 2005.

[18] A. D. Briggs, O. T. Mytton, A. Kehlbacher, R. Tiffin, M. Rayner, and P. Scarborough. Overall and income specific effect on prevalence of overweight and obesity of 20% sugar sweetened drink tax in uk: Econometric and comparative risk assessment modelling study. *BMJ*, 347:f6189, 2013.

[19] J. Y. Chai, M. Budzikowska, V. Horvath, N. Nicolov, N. Kambhatla, and W. Zadrozny. Natural Language Sales Assistant-A Web-Based Dialog System for Online Sales. In *Innovative Applications of Artificial Intelligence*, pages 19–26, 2001.

[20] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[21] R. B. Clariana and R. Koul. A computer-based approach for translating text into concept map-like representations. In *Proceedings of the First Int. Conference on Concept Mapping*, 2004.

[22] F. Colas, P. Paclík, J. N. Kok, and P. Brazdil. Does SVM really scale up to large bag of words feature spaces? In *International Symposium on Intelligent Data Analysis*, pages 296–307. Springer, 2007.

[23] C. Collins, S. Carpendale, and G. Penn. DocuBurst: Visualizing document content using language structure. *Computer Graphics Forum*, 28(3):1039–1046, 2009.

[24] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. Association for Computing Machinery, 2008.

[25] R. Crutzen and P. Giabbanelli. Using classifiers to identify binge drinkers based on drinking motives. *Substance use & misuse*, 49(1–2):110–115, 2014.

[26] R. Crutzen, P. Giabbanelli, A. Jander, L. Mercken, and H. de Vries. Identifying binge drinkers based on parenting dimensions and alcohol-specific parenting practices: Building classifiers on adolescent-parent paired data. *BMC Public Health*, 15(1):747, 2015.

[27] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu. Context preserving dynamic word cloud visualization. In *2010 IEEE Pacific Visualization Symposium (PacificVis)*, pages 121–128. IEEE, 2010.

[28] J. Dickerson, Z. Cox, E. Wurtele, and A. Fulmer. Creating metabolic and regulatory network models using fuzzy cognitive maps. In *IFSA World Congress and 20th North American*

Fuzzy Information Processing Society International Conference, 2001. Joint 9th, volume 4, pages 2171–2176. IEEE, 2001.

[29] R. Diestel. *Graph Theory - Graduate Texts in Mathematics*, pages 2,25. Springer-Verlag Berlin and Heidelberg GmbH & amp, 1 edition, 2000.

[30] E. M. Douglas, S. A. Wheeler, D. J. Smith, I. C. Overton, S. A. Gray, T. M. Doody, and N. D. Crossman. Using mental-modelling to explore how irrigators in the Murray–Darling Basin make water-use decisions. *Journal of Hydrology: Regional Studies*, 6:1 – 12, 2016.

[31] N. M. Dowell, O. Skrypnyk, S. Joksimovic, A. C. Graesser, S. Dawson, D. Gaševic, T. A. Hennis, P. de Vries, and V. Kovanovic. Modeling learners' social centrality and performance through language and discourse. *International Educational Data Mining Society*, 2015.

[32] M. Dredze, R. Cheng, M. Paul, and D. Broniatowski. Healthtweets.org: A platform for public health surveillance using Twitter. *Association for the Advancement of Artificial Intelligence Workshop on the World Wide Web and Public Health Intelligence*, 2014.

[33] A. Endert, R. Burtner, N. Cramer, R. Perko, S. Hampton, and K. Cook. Typograph: Multiscale spatial exploration of text documents. In *2013 IEEE International Conference on Big Data*, pages 17–24, 2013.

[34] G. Erkan and D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479, Dec. 2004.

[35] H. Eyles, C. N. Mhurchu, N. Nghiem, and T. Blakely. Food pricing strategies, population diets, and non-communicable disease: A systematic review of simulation studies. *PLoS Medicine*, 9(12):e1001353, 2012.

[36] R. Eynon, I. Hjorth, T. Yasseri, and N. Gillani. Understanding communication patterns in MOOCs: Combining data mining and qualitative methods. *arXiv preprint arXiv:1607.07495*, 2016.

[37] J. Fan and Y. Fan. High dimensional classification using features annealed independence rules. *Annals of statistics*, 36(6):2605, 2008.

[38] M. Fischer. The KWIC index concept: A retrospective view. *Journal of the Association for Information Science and Technology*, 17(2):57–70, 1966.

[39] E. A. Fox. *Extending the Boolean and Vector Space Models of Information Retrieval with P-norm Queries and Multiple Concept Types*. PhD thesis, Cornell University, Ithaca, NY, USA, 1983. AAI8328584.

[40] R. Freedman. Plan-based dialogue management in a physics tutor. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLP '00, pages 52–59, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[41] R. Freedman. Using a reactive planner as the basis for a dialogue agent. In *Proceedings of Florida Artificial Intelligence Research Society Conference 2000*, pages 203–208, 2000.

[42] N. Fuhr. Probabilistic models in information retrieval. *The Computer Journal*, 35(3):243–255, 1992.

[43] P. Gambette and J. Véronis. Visualising a text with a tree cloud. In *Classification as a Tool for Research*, pages 561–569. Springer, 2010.

[44] D. García-Saiz, C. Palazuelos, and M. Zorrilla. *Educational Data Mining: Applications and Trends*, chapter Data Mining and Social Network Analysis in the Educational Field: An Application for Non-Expert Users, pages 411–439. Springer, Cham, 2014.

[45] W. Gasarch. Chapter 16 a survey of recursive combinatorics. In *Handbook of Recursive MathematicsVolume 2: Recursive Algebra, Analysis and Combinatorics*, volume 139 of *Studies in Logic and the Foundations of Mathematics*, pages 1041 – 1176. Elsevier, 1998.

[46] D. Gasevic, V. Kovanovic, S. Joksimovic, and G. Siemens. Where is research on massive open online courses headed? A data analysis of the MOOC research initiative. *The International Review of Research in Open and Distributed Learning*, 15(5), 2014.

[47] P. Giabbanelli. The small-world property in networks growing by active edges. *Advances in Complex Systems*, 14(6):853–869, 2011.

[48] P. Giabbanelli and J. Adams. Identifying small groups of foods that can predict achievement of key dietary recommendations: Data mining of the uk national diet and nutrition survey, 2008–12. *Public Health Nutrition*, 19(9):1543–1551, 2016.

[49] P. Giabbanelli and J. Peters. Réseaux complexes et épidemies. *Technique et science informatiques*, 30(2):181–212, 2011.

[50] P. J. Giabbanelli. Modelling the spatial and social dynamics of insurgency. *Security Informatics*, 3(1):1–15, 2014.

[51] P. J. Giabbanelli, J. Adams, and V. S. Pillutla. Feasibility and framing of interventions based on public support: Leveraging text analytics for policymakers. In G. Meiselwitz, editor, *Social Computing and Social Media: 8th International Conference, SCSM 2016, Held as Part of HCI International 2016, Toronto, ON, Canada, July 17–22, 2016. Proceedings*, pages 188–200. Springer, 2016.

[52] P. J. Giabbanelli and R. Crutzen. Creating groups with similar expected behavioural response in randomized controlled trials: A fuzzy cognitive map approach. *BMC medical research methodology*, 14(1):130, 2014.

[53] P. J. Giabbanelli, P. J. Jackson, and D. T. Finegood. Modelling the joint effect of social determinants and peers on obesity among canadian adults. In *Theories and simulations of complex social systems*, pages 145–160. Springer, 2014.

[54] P. J. Giabbanelli, P. J. Jackson, and D. T. Finegood. *Theories and Simulations of Complex Social Systems*, chapter Modelling the Joint Effect of Social Determinants and Peers on Obesity Among Canadian Adults, pages 145–160. 2014.

[55] P. J. Giabbanelli, T. Torsney-Weir, and V. K. Mago. A fuzzy cognitive map of the psychosocial determinants of obesity. *Applied soft computing*, 12(12):3711–3724, 2012.

[56] N. Gillani and R. Eynon. Communication patterns in massively open online courses. *The Internet and Higher Education*, 23:18–26, 2014.

[57] M. Glykas. Fuzzy cognitive maps. *Studies in Fuzziness and Soft Computing*, 2010.

[58] S. Goggins, K. Galyen, E. Petakovic, and J. Laffey. Connecting performance to social structure and pedagogy as a pathway to scaling learning analytics in MOOCs: An exploratory study. *Journal of Computer Assisted Learning*, 32(3):244–266, 2016.

[59] P. Goodyear, C. Jones, and K. Thompson. *Computer-Supported Collaborative Learning: Instructional Approaches, Group Processes and Educational Designs*, pages 439–451. Springer, New York, NY, 2014.

[60] C. Goutte and E. Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European Conference on Information Retrieval*, pages 345–359. Springer, 2005.

[61] A. C. Graesser, D. S. McNamara, and J. M. Kulikowich. Coh-Metrix providing multilevel analyses of text characteristics. *Educational researcher*, 40(5):223–234, 2011.

[62] S. Gray, S. Gray, J. L. D. Kok, A. E. Helfgott, B. O'Dwyer, R. Jordan, and A. Nyaki. Using fuzzy cognitive mapping as a participatory approach to analyze change, preferred states, and perceived resilience of social-ecological systems. *Ecology and Society*, 20(2):11, 2015.

[63] A. Gregg. *Online Graduate Students' Experiences with Asynchronous Course Discussions*. PhD thesis, The Pennsylvania State University, 2016.

[64] B. Gretarsson, J. O'donovan, S. Bostandjiev, T. Höllerer, A. Asuncion, D. Newman, and P. Smyth. Topicnets: Visual analysis of large text corpora with topic modeling. *Transactions on Intelligent Systems and Technology (TIST)*, 3(2):23, 2012.

[65] A. Guille and C. Favre. Event detection, tracking, and visualization in Twitter: a mention-anomaly-based approach. *Social Network Analysis and Mining*, 5(1):1–18, 2015.

[66] C. N. Gunawardena, C. A. Lowe, and T. Anderson. Analysis of a global online debate and the development of an interaction analysis model for examining social construction of knowledge in computer conferencing. *Journal of educational computing research*, 17(4):397–431, 1997.

[67] D. Guo, J. Chen, A. M. MacEachren, and K. Liao. A visualization system for space-time and multivariate patterns (vis-stamp). *IEEE transactions on visualization and computer graphics*, 12(6):1461–1474, 2006.

[68] C. Gutwin. Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 267–274. ACM, 2002.

[69] R. Hamad, J. L. Pomeranz, A. Siddiqi, and S. Basu. Large-scale automated analysis of news media: A novel computational method for obesity policy research. *Obesity*, 23(2):296–300, 2014.

[70] H. He, Y. Bai, E. A. Garcia, and S. Li. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328. IEEE, 2008.

[71] J. Henderson. Neural network probability estimation for broad coverage parsing. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 131–138. Association for Computational Linguistics, 2003.

[72] K. F. Hew and W. S. Cheung. Students' and instructors' use of massive open online courses (MOOCs): Motivations and challenges. *Educational Research Review*, 12:45–58, 2014.

[73] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 2012.

[74] G. Hirst and D. St-Onge. Lexical chains as representations of context for the detection and correction of malapropisms. *WordNet: An electronic lexical database*, 305:305–332, 1998.

[75] M. Hu, K. Wongsuphasawat, and J. Stasko. Visualizing social media content with sententree. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):621–630, 2017.

[76] J. Huang, M. Zhou, and D. Yang. Extracting chatbot knowledge from online discussion forums. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 423–428, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers.

[77] J. L. Hutchens and M. D. Alder. Introducing MegaHAL. In *Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning*, NeMLaP3/CoNLL '98, pages 271–274, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

[78] N. Japkowicz. Class imbalances: Are we focusing on the right issue. In *Workshop on Learning from Imbalanced Data Sets II*, volume 1723, page 63, 2003.

[79] X. Ji, S. A. Chun, and J. Geller. Monitoring public health concerns using Twitter sentiment classifications. In *Healthcare Informatics (ICHI), 2013 IEEE International Conference on*, pages 335–344. IEEE, 2013.

[80] X. Ji, S. A. Chun, Z. Wei, and J. Geller. Twitter sentiment classification for measuring public health concerns. *Social Network Analysis and Mining*, 5(1):1–25, 2015.

[81] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[82] P. Jordan, M. Ringenberg, and B. Hall. Rapidly developing dialogue systems that support learning studies. In *Proceedings of Intelligent Tutoring Systems'06 Workshop on Teaching with Robots, Agents, and NLP*, pages 1–8, 2006.

[83] V. Jordão, D. Gonçalves, and S. Gama. EduVis: visualizing educational information. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, pages 1011–1014. ACM, 2014.

[84] J. Kim, M. Han, Y. Lee, and Y. Park. Futuristic data-driven scenario building: Incorporating text mining and fuzzy association rule mining into fuzzy cognitive map. *Expert Systems with Applications*, 57:311–323, 2016.

[85] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[86] G. J. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, Upper Saddle River, NJ, USA, 1995. pages 1-48.

[87] F. KM, C. MD, O. CL, and C. LR. Prevalence and trends in obesity among us adults, 1999-2008. *Journal of the American Medical Association (JAMA)*, 303(3):235–241, 2010.

[88] D. Koller, A. Ng, C. Do, and Z. Chen. Retention and intention in massive open online courses: In depth. *Educause Review*, 48(3):62–63, 2013.

[89] B. Kosko. Fuzzy cognitive maps. *International Journal of man-machine studies*, 24(1):65–75, 1986.

[90] W. Kraaij and M. Spitters. *Language Models for Topic Tracking*, pages 95–123. Springer, Dordrecht, 2003.

[91] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[92] L. I. Kuncheva. *Combining pattern classifiers: Methods and algorithms*, pages 9–12. John Wiley & Sons, 2004.

[93] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM Special Interest Group on Information Retrieval conference on Research and development in information retrieval*, pages 111–119. ACM, 2001.

[94] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. In *Language modeling for information retrieval*, pages 1–10. Springer, 2003.

[95] W. Liu, Ł. Kidziński, and P. Dillenbourg. *Semiautomatic Annotation of MOOC Forum Posts*, pages 399–408. Springer Singapore, Singapore, 2016.

[96] H. P. Luhn. A new method of recording and searching information. *American Documentation (pre-1986)*, 4(1):14–16, 1953.

[97] L. F. Luna-Reyes and D. L. Andersen. Collecting and analyzing qualitative data for system dynamics: Methods and models. *System Dynamics Review*, 19(4):271–296, 2003.

[98] A. Madani, O. Boussaid, and D. E. Zegour. Real-time trending topics detection and description from twitter content. *Social Network Analysis and Mining*, 5(1):59, 2015.

[99] A. Makazhanov, D. Rafiei, and M. Waqar. Predicting political preference of twitter users. *Social Network Analysis and Mining*, 4(1):1–15, 2014.

[100] E. H. Mamdani. Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE transactions on computers*, 100(12):1182–1191, 1977.

[101] T. Mandl. Das COSIMIR-Modell für information retrieval mit neuronalen Netzen. *Datenbank Rundbrief*, 24:54–60, 1999.

[102] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999, pages 268–269. MIT Press, 1999.

[103] J. D. Mcauliffe and D. M. Blei. Supervised topic models. In *Advances in neural information processing systems*, pages 121–128, 2008.

[104] M. F. McTear. Spoken dialogue technology: Enabling the conversational user interface. *ACM Computing Surveys (CSUR)*, 34(1):90–169, Mar. 2002.

[105] D. H. Meadows. *Thinking in Systems: A Primer*. Chelsea Green, 2008.

[106] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. In D. Lin and D. Wu, editors, *Proceedings of Empirical Methods on Natural Language Processing 2004*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[107] R. Mollineda, R. Alejo, and J. Sotoca. The class imbalance problem in pattern classification and learning. In *II Congreso Español de Informática (CEDI 2007). ISBN*, pages 978–84, 2007.

[108] B. M. Moon, R. R. Hoffman, J. D. Novak, and A. J. Canas. *Applied Concept Mapping: Capturing, Analyzing, and Organizing Knowledge*. CRC Press, 2011.

[109] R. Navigli and M. Lapata. Graph connectivity measures for unsupervised word sense disambiguation. In *International Joint Conference on Artificial Intelligence*, pages 1683–1688, 2007.

[110] L. Nixon, P. Mejia, A. Cheyne, and L. Dorfman. Big Soda's long shadow: News coverage of local proposals to tax sugar-sweetened beverages in Richmond, El Monte and Telluride. *Critical Public Health*, 25(3):333–347, 2015.

[111] J. Otterbacher, G. Erkan, and D. R. Radev. Biased LexRank: Passage retrieval using random walks with question-based priors. *Information Processing and Management*, 45(1):42 – 54, 2009.

[112] C. D. Paice. Constructing literature abstracts by computer: techniques and prospects. *Information Processing & Management*, 26(1):171–186, 1990.

[113] E. I. Papageorgiou and J. L. Salmeron. A review of fuzzy cognitive maps research during the last decade. *IEEE Transactions on Fuzzy Systems*, 21(1):66–79, 2013.

[114] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[115] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM Special Interest Group on Information Retrieval conference on Research and development in information retrieval*, pages 275–281. ACM, 1998.

[116] N. Poolsawad, C. Kambhampati, and J. Cleland. Balancing class for performance of classification with a clinical dataset. In *Proceedings of the World Congress on Engineering*, volume 1, 2014.

[117] T. M. Pope. Limiting liberty to prevent obesity: Justifiability of strong hard paternalism in public health regulation. *Connecticut Law Review*, 46:1859–1876, 2013.

[118] S. F. Pratt, P. J. Giabbanelli, and J.-S. Mercier. Detecting unfolding crises with visual analytics and conceptual maps emerging phenomena and big data. In *2013 IEEE International Conference on Intelligence and Security Informatics*, pages 200–205. IEEE, 2013.

[119] R. Rabbany, S. Elatia, M. Takaffoli, and O. R. Zaïane. *Educational Data Mining: Applications and Trends*, chapter Collaborative Learning of Students in Online Discussion Forums: A Social Network Analysis Perspective, pages 441–466. Springer, 2014.

[120] D. R. Radev, E. Hovy, and K. McKeown. Introduction to the special issue on summarization. *Computational linguistics*, 28(4):399–408, 2002.

[121] M. M. Rahman and D. Davis. Cluster based under-sampling for unbalanced cardiovascular data. In *Proceedings of the World Congress on Engineering*, volume 3, pages 3–5, 2013.

[122] K. Rajaraman and A. Tan. Knowledge discovery from texts: A concept frame graph approach. In *Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 669–671, 2002.

[123] J. S. Risch, R. A. May, S. T. Dowson, and J. J. Thomas. A virtual environment for multimedia intelligence data analysis. *IEEE Computer Graphics and Applications*, 16(6):33–41, 1996.

[124] C. Rivard, D. Smith, S. E. McCann, and A. Hyland. Taxing sugar-sweetened beverages: A survey of knowledge, attitudes and behaviours. *Public Health Nutrition*, 15(8):1355–1361, 2012.

[125] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th annual international ACM Special Interest Group on Information Retrieval conference on Research and development in information retrieval*, pages 232–241. Springer-Verlag New York, 1994.

[126] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. *NIST SPECIAL PUBLICATION SP*, 109:109, 1995.

[127] S. Robinson. *Simulation: The practice of model development and use*, pages 184–186. 2 edition, 2016.

[128] L. A. Rossi and O. Gnawali. Language independent analysis and classification of discussion threads in coursera mooc forums. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*, pages 654–661. IEEE, 2014.

[129] A. P. Rovai. Facilitating online discussions effectively. *The Internet and Higher Education*, 10(1):77–88, 2007.

[130] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[131] P. Sacher and C. Swain. The mend programme: Tackling childhood obesity. *British Journal of School Nursing*, 2:4, 2007.

[132] H. Sak, A. W. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH*, pages 338–342, 2014.

[133] M. Salathé and S. Khandelwal. Assessing vaccination sentiments with online social media: Implications for infectious disease dynamics and control. *PLOS Computational Biology*, 7(10):1–7, 10 2011.

[134] G. Salton. The smart retrieval system: Experiments in automatic document processing. *IEEE Transactions on Professional Communication*, PC-15(1):17–17, March 1972.

[135] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[136] G. Salton, A. Singhal, M. Mitra, and C. Buckley. Automatic text structuring and summarization. *Information Processing & Management*, 33(2):193–207, 1997.

[137] G. Salton, A. Wong, and C.-S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[138] Linear SVC, Scikit-learn: Machine learning in Python, 2016. URL: `http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC` (Visited on 08/26/2016).

[139] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136, 2011.

[140] P. Soucy and G. W. Mineau. Beyond tf-idf weighting for text categorization in the vector space model. In *International Joint Conference on Artificial Intelligence*, volume 5, pages 1130–1135, 2005.

[141] R. Speer, C. Havasi, N. Treadway, and H. Lieberman. Visualizing common sense connections with luminoso. *Proceedings of the first international workshop on Intelligent visual interfaces for text analysis*, pages 9–12, 2010.

[142] J. Stasko, C. Görg, and Z. Liu. Jigsaw: Supporting investigative analysis through interactive visualization. *Information visualization*, 7(2):118–132, 2008.

[143] P. Stoddard, B. van Erp, M. Induni, A. Reedy, and B. Broderick. Sugar-sweetened beverage tax acceptability in the u.s. versus foreign-born population: Results from a large Northern California county. *142nd Annual Meeting & Exposition of the American Public Health Association (APHA)*, page 303809, 2014.

[144] M.-A. D. Storey and H. A. Müller. Graph layout adjustment strategies. In *International Symposium on Graph Drawing*, pages 487–499. Springer, 1995.

[145] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[146] K. Swan and L. F. Shih. On the nature and development of social presence in online course discussions. *Journal of Asynchronous learning networks*, 9(3):115–136, 2005.

[147] The People of the City of Berkeley. Imposing a general tax on the distribution of sugar-sweetened beverage products. *Ordinance*, 2014.

[148] Y.-H. Tseng, C.-Y. Chang, S.-N. C. Rundgren, and C.-J. Rundgren. Mining concept maps from news stories for measuring civic scientific literacy in media. *Computers & Education*, 55(1):165 – 177, 2010.

[149] K. Vanlehn, P. W. Jordan, C. P. Rose, D. Bhembe, M. Boettner, A. Gaydos, M. Makatchev, U. Pappuswamy, M. Ringenberg, A. Roque, S. Siler, and R. Srivastava. The architecture of

Why2-Atlas: A coach for qualitative physics essay writing. In *In Proceedings of Intelligent Tutoring Systems Conference, volume 2363 of LNCS*, pages 158–167. Springer, 2002.

[150] T. Verigin, P. J. Giabbanelli, and P. Davidsen. Supporting a systems approach to healthy weight interventions in british columbia by modeling weight and well-being. *Proceedings of the 2016 Annual Simulation Symposium (ANSS)*, 2016.

[151] J. Villalon and R. A. Calvo. Concept maps as cognitive visualizations of writing assignments. *Educational Technology & Society*, 14(3):16–27, 2011.

[152] O. Vinyals and Q. V. Le. A neural conversational model. *CoRR*, abs/1506.05869, 2015.

[153] F. Y. Wang, K. M. Carley, D. Zeng, and W. Mao. Social computing: From social informatics to social intelligence. *IEEE Intelligent Systems*, 22(2):79–83, March 2007.

[154] W. M. Wang, C. F. Cheung, W. Lee, and S. Kwok. Mining knowledge from natural language texts using fuzzy associated concept mapping. *Information Processing & Management*, 44(5):1707–1719, 2008.

[155] M. Wattenberg and F. B. Viégas. The word tree, an interactive visual concordance. *IEEE transactions on visualization and computer graphics*, 14(6):1221–1228, 2008.

[156] A. F. Wise and M. M. Chiu. Analyzing temporal patterns of knowledge construction in a role-based online discussion. *International Journal of Computer-Supported Collaborative Learning*, 6(3):445–470, 2011.

[157] P. C. Wong, B. Hetzler, C. Posse, M. Whiting, S. Havre, N. Cramer, A. Shah, M. Singhal, A. Turner, and J. Thomas. IN-SPIRE InfoVis 2004 contest entry. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages r2–r2. IEEE, 2004.

[158] Z. Xu and Q.-L. Da. An overview of operators for aggregating information. *International Journal of Intelligent Systems*, 18(9):953–969, 2003.

[159] D. Yang, T. Sinha, D. Adamson, and C. P. Rosé. Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses. In *Proceedings of the 2013 NIPS Data-driven education workshop*, volume 11, page 14, 2013.

[160] C. Zhai. Statistical language models for information retrieval. *Synthesis Lectures on Human Language Technologies*, 1(1):1–37, 2008.

[161] C. Zhai. Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–213, Mar. 2008.

[162] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410. ACM, 2001.

[163] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM Special Interest Group on Information Retrieval conference on Research and development in information retrieval*, pages 334–342. ACM, 2001.

[164] D. Zhang, P. J. Giabbanelli, O. A. Arah, and F. J. Zimmerman. Impact of different policies on unhealthy dietary behaviors in an urban adult population: an agent-based simulation model. *American journal of public health*, 104(7):1217–1222, 2014.