FLORIDA POLYTECHNIC UNIVERSITY

THESIS TITLE GOES HERE

by

Levi C. Nicklas

A Thesis Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

Florida Polytechnic University

2021

THESIS TITLE GOES HERE

by

Levi C. Nicklas

A Thesis Submitted to the Faculty of the

DEPARTMENT OF COMPUTER SCIENCE

In Partial Fulfillment of the Requirements

For the Degree of

MASTER OF SCIENCE

In the Graduate College

Florida Polytechnic University

2021

Approved by:

	Signature	**Date**
Dr. Reinaldo Sanchez-Arias (Committee Chair, Advisor)	_____	_____
Dr. Grisselle Centeno (Committee Member)	_____	_____
Dr. Harish Chintakunta (Committee Member)	_____	_____
Dr. Tom Dvorske Vice Provost of Academic Affairs (Graduate Division)	_____	_____

Dedications go here

ACKNOWLEDGMENTS

Acknowledgments go here

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

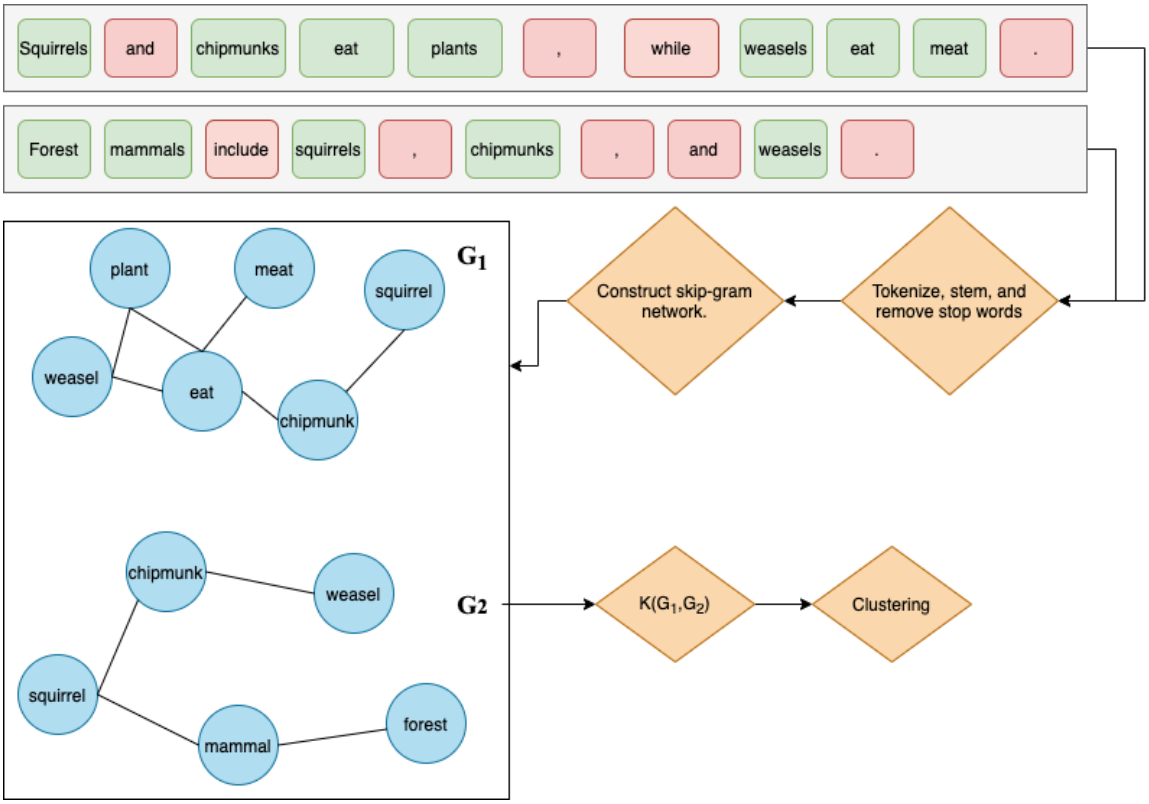
Abstract goes here

Chapter 1: Introduction

Introduction Chapter Here

Text mining is the process of extracting insight from text documents using computational and statistical methods. A common task in text mining is document clustering, where the natural language of the documents is reformatted and used to group documents into clusters of similar topics or text content. Popular methods to cluster text include k-means, hierarchical methods, and topic models. Most of these methods utilize term frequency-inverse document frequency measures or bag-of-words methods to do clustering, though these methods often divide words and thus may lose meaning, or context, surrounding the word of concern. In an effort to cluster documents while still preserving the relationship between words, text can be modeled with graphs which better preserve the context surrounding a word. Text can be represented with bigram graphs, where the edge between two vertices is the result of two words appearing adjacent in the text. This idea can be extended to more general n-grams, and further extended to skip-n-grams, resulting in even richer representations of the text. The similarity of two of graphical representations can then be assessed using modern methods called graph kernels. Graph kernels produce a measure of similarity between graphs, thus allowing for further machine learning to take place. With a measure of similarity between two graphs, we can do an assortment of clustering methods.

The chosen graph representation of the observational unit of text is an extension of bigram graphs. The use of skip-grams here is an attempt to connect ideas between words that would have normally attained meaning through the "context" of the sentence. By connecting words that appear close to one another while not being immediately adjacent, we are creating "skip-grams"– an extension of the bigram where the window for which words are considered adjacent is widened.



In the above figure, a demonstration of bigram construction is available (**Levi**: This graphic needs to be a skipgram for which $k > 1$, and comments thereafter need to be about the skip gram set up...this will be a place holder for now).

Many methods used in text mining use bag-of-words techniques, but these methods do not always preserve the context and relationships between words. The graph representations for text, used here, aim to preserve the relationship and context between words. The idea of bigrams (pairs of two words) can be extended to "skip-grams"; skip-grams are words which appear within some width w of each other. This leads to more connections, and a richer graph representation of the text. This skip-gram method is similar to that used in the popular machine learning package Word2Vec, which uses the "continuous bag of words" or "continuous skip-grams". In

this paper, a graph representation of the text is constructed, using skip-gram methods, for each comment in the dataset.

The graph representations are compared with a graph kernel, which is a measure of the similarity of two graphs. In this study, the graph kernel of choice is a "edge histogram kernel", because the kernel utilizes the labels of the vertices to assess the similarity of two graphs– not just the topology. For each graph, the kernel produces a measure of similarity to the other n-1 graphs in the dataset. We can use this similarity measure between two graphs, to assess how similar other graphs in the dataset may be. That is, we can compare the similarity of graphs B and C through their similarity to A. To do this, a KDE (Kernel Density Estimation) to estimate a kernel density curve, the curve is then used to partition the rest of the dataset into clusters. The KDE bandwidth is modified to produce more or less local minima/maxima that can be then used to identify potential clusters. Basic calculus can be used to locate to inflection points that will serve as intervals for which a cluster will be defined.

Alternatively, we can use machine learning methods which work nicely with kernels, i.e. support vector machine (supervised) or hierarchical clustering (unsupervised). These methods are more tried and true, and can serve as a comparison to the success of the KDE methods.

To test these methods, two different text data sets will be considered. First, reddit comments from subreddits pertaining to mental health. Using the redditExtractoR package by (NAME), the comments are collected if they match criteria set by the query. From the subreddit r/mentalhealth, posts are collected if they have chosen keywords and at least 10 comments. The keywords considered are "anxious", "anxiety", ""depressed", "depression", "mental", "illness", "scared", "afraid", "sad", "emotion", "anger", "angry", "upset", "suicide", "abuse", "emotional", "help", and "addiction". These words do not represent all words which define or indicate mental health topics

may be present; these words were chosen by the researcher arbitrarily. The result of the queries is thousands of comments. Mental health continues to be a growing focus at the national level, and the discussion on reddit is often candid and open, due to reddit's reputation as an anonymous website. The second data set being considered in the study, is NHTSA report data. These reports are form the Special Crash Investigation (SCI) reports from the NHTSA (National Highway Traffic Safety Administration). The pdf documents that are considered in this study are those that involved ambulance(s) in the incidents highlighted in the reports. These events are considered "edge cases" and could prove useful to those working in autonomous vehicles.

These datasets were chosen because they are two very different styles of writing. The reddit posts are often in contemporary and casual use of language, possibly including "netspeak", while the NHTSA reports feature a technical writing style with plain language, absent of hyperbole or sarcasm. These datasets should show two very different sides of text data, and comparisons will be made to how each dataset responds to the methods outlined above.

Chapter 2: Literature Review

Literature review chapter goes here

2.1 Introduction

Introduction section of this chapter goes here

2.1.1 Subsection Testing

subection content goes here

Chapter 3: Methods

3.1 Introduction

3.1.1 Skip-grams

As an alternative to natural language processing (NLP) methods, which are reliant on "bag-of-words" methods, the methods used here utilize a graph representation of the text. Consider a bigram, a pair of two words—like "hot dog" or "peanut butter", these bigrams can be constructed for a text document where every pair or adjacent words is a bigram. The bigrams can then be used to make a graph, where each word is a vertex, and each bigram is an edge. This graph representation holds more context than the bag-of-words methods; for example seeing the words "cake" and "carrot" in a bag of words may not show that "carrot cake" was the real intent of the text. This is an important concept for modeling text, as we should strive to achieve a representation of the text that makes for effective modeling that will capture the true meaning of the text in question. Keeping this in mind, with the example of "carrot cake", what about the idiom "beating a dead horse"? Each word individually may mean something other than the idiom. Even the bigrams "beating dead" and "dead horse" do not capture what the idiom means. We can expand the number of words in the n-gram to be 3 or 4 words, or alternatively, we can make more "edges" or connect more words. We can connect words that are not immediately adjacent but perhaps within k words away. These bigrams that appear within k words of each other are called "skip-grams". The skip-gram allows to capture context of larger sequences of words since the graph representation will show how the k wide neighborhood of words was connected. In the idiom example, using skip-grams with window width $k = 2$, and removing common words (e.g. "a", "at", "the"), will produce a graph like:

$$E(G) = \{\text{beat} \longleftrightarrow \text{dead}, \text{dead} \longleftrightarrow \text{horse}, \text{horse} \longleftrightarrow \text{beat}\}$$

This graph representation contains a cycle, of length 3, where most native english speakers will identify the meaning behind the graph representation. As ideas, idioms, figures of speech, and other concepts (that may be explained in a non-literal fashion) grow in size as they include more words, it becomes more difficult to capture the meaning behind the text. However, leveraging the concept of a skip gram can produce such a rich graph representation of the text that the original meaning is more likely to be preserved.

3.1.2 Graph Kernels

The next natural question is, "how can we compare these graph representations?", and we address this with graph kernel methods. These methods are used to compare the similarity of graphs. These use of a graph kernel to compare graphs was first published in 2003, and since various applications and adaptations have been made to the methods. In the case of text mining, the graph kernel must assess vertex labels —if one intends to map words to vertices, otherwise they will be assessing the topology alone. In this study, the Edge-Histogram kernel is the kernel used to compute similarity. This kernel was chosen as it uses labels on the graph structure, and is not as computationally intensive as other methods (CITE). In the specific implementation used for these studies, the computation time was the shortest when compared with other kernel methods like: graphlet, random walk, and Weisfeiler-Lehman kernel. (CITE) Since the data sets of concern in the studies feature either large graphs or a large number of graphs, the kernel had to be cheap computationally.

To compute and edge histogram kernel on two graphs, G_1 and G_2, first define the set of edges $E_i = \{(u_1, v_1), (u_2, v_2), ..., (u_n, v_n)\}$ where (u_n, v_n) is the n-th edge connecting u_n to v_n. Then the edge label histogram is defined to be $\vec{g} = \{g_1, g_2, ..., g_s\}$ where (FINISH

THIS, link in comment)

3.1.3 Using Kernel for Clustering

The output of the kernel is useful for a variety of tasks. Some other popular applications have included classification with support vector machines, which are popular with other kernel methods. In this case, the kernel is used for unsupervised clustering. Within the kernel matrix, K, the entry $k_{i,j}$ represents the similarity between graphs i and j. This matrix which contains measures of similarity between points can be used as a distance matrix for hierarchical clustering. Before using the graph kernel as a distance matrix, normalization or standardization takes place, and principal component analysis may be used. The end result is each row is a single graph-document being described by its similarity to all the other graphs, which are the column values. Once the values are transformed or rotated by preprocessing methods, the points are just represented by their similarity to one another, but in a transformed space. Various hyper parameters can be tuned for successful clustering; the graph kernel has a parameter that can be tuned, and the hierarchical clustering can be tried with differing types of linkage.

3.1.4 Kernel Density Estimation Clustering for Linear Kernel

Rest of sections go here

Chapter 4: Discussion of results

Discussions and results go here

Chapter 5: Conclusion

Conclusion goes here.

Chapter 6: Future Work

Future work goes here.

Appendices

Appendix A: My first appendix