

University of Wisconsin Milwaukee

## UWM Digital Commons

---

### Theses and Dissertations

---

December 2022

# Use of Machine Learning and Natural Language Processing to Enhance Traffic Safety Analysis

MD ABU SAYED

*University of Wisconsin-Milwaukee*

Follow this and additional works at: <https://dc.uwm.edu/etd>



Part of the [Civil Engineering Commons](#), and the [Computer Sciences Commons](#)

---

### Recommended Citation

SAYED, MD ABU, "Use of Machine Learning and Natural Language Processing to Enhance Traffic Safety

Analysis" (2022). *Theses and Dissertations*. 3071.

<https://dc.uwm.edu/etd/3071>

This Dissertation is brought to you for free and open access by UWM Digital Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UWM Digital Commons. For more information, please contact [scholarlycommunicationteam-group@uwm.edu](mailto:scholarlycommunicationteam-group@uwm.edu).

USE OF MACHINE LEARNING AND NATURAL LANGUAGE PROCESSING  
TO ENHANCE TRAFFIC SAFETY ANALYSIS

by

Md Abu Sayed

A Dissertation Submitted in

Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

in Engineering

at

The University of Wisconsin-Milwaukee

December 2022

## ABSTRACT

# USE OF MACHINE LEARNING AND NATURAL LANGUAGE PROCESSING TO ENHANCE TRAFFIC SAFETY ANALYSIS

by

Md Abu Sayed

The University of Wisconsin-Milwaukee, December 2022  
Under the Supervision of Professor Xiao Qin

Despite significant advances in vehicle technologies, safety data collection and analysis, and engineering advancements, tens of thousands of Americans die every year in motor vehicle crashes. Alarmingly, the trend of fatal and serious injury crashes appears to be heading in the wrong direction. In 2021, the actual rate of fatalities exceeded the predicted rate. This worrisome trend prompts and necessitates the development of advanced and holistic approaches to determining the causes of a crash (particularly fatal and major injuries). These approaches range from analyzing problems from multiple perspectives, utilizing available data sources, and employing the most suitable tools and technologies within and outside traffic safety domain.

The primary source for traffic safety analysis is the structure (also called tabular) data collected from crash reports. However, structure data may be insufficient because of missing information, incomplete sequence of events, misclassified crash types, among many issues. Crash narratives, a form of free text recorded by police officers to describe the unique aspects and circumstances of a crash, are commonly used by safety professionals to supplement structure data fields. Due to its unstructured nature, engineers have to manually review every crash narrative. Thanks to the rapid development in natural language processing (NLP) and machine

learning (ML) techniques, text mining and analytics has become a popular tool to accelerate information extraction and analysis for unstructured text data. The primary objective of this dissertation is to discover and develop necessary tools, techniques, and algorithms to facilitate traffic safety analysis using crash narratives.

The objectives are accomplished in three areas: enhancing data quality by recovering missed crashes through text classification, uncovering complex characteristics of collision generation through information extraction and pattern recognition, and facilitating crash narrative analysis by developing a web-based tool. At first, a variety of NoisyOR classifiers were developed to identify and investigate work zone (WZ), distracted (DD), and inattentive (ID) crashes. In addition, various machine learning (ML) models, including multinomial naive bayes (MNB), logistic regression (LGR), support vector machine (SVM), k-nearest neighbor (K-NN), random forest (RF), and gated recurrent unit (GRU), were developed and compared with NoisyOR. The comparison shows that NoisyOR is simple, computationally efficient, theoretically sound, and has one of the best model performances. Furthermore, a novel neural network architecture named Sentence-based Hierarchical Attention Network (SHAN) was developed to classify crashes and its performance exceeds that of NoisyOR, GRU, Hierarchical Attention Network (HAN), and other ML models. SHAN handled noisy or irrelevant parts of narratives effectively and the model results can be visualized by attention weight.

Because a crash often comprises a series of actions and events, breaking the chain of events could prevent a crash from reaching its most dangerous stage. With the objectives of creating crash sequences, discovering pattern of crash events, and finding missing events, the Part-of-Speech tagging (PT), Pattern Matching with POS Tagging (PMPT), Dependency Parser

(DP), and Hybrid Generalized (HGEN) algorithms were developed and thoroughly tested using crash narratives. The top performer, HGEN, uses predefined events and event-related action words from crash narratives to find new events not captured in the data fields. Besides, the association analysis unravels the complex interrelations between events within a crash.

Finally, the crash information extraction, analysis, and classification tool (CIEACT), a simple and flexible online web tool, was developed to analyze crash narratives using text mining techniques. The tool uses a Python-based Django Web Framework, HTML, and a relational database (PostgreSQL) that enables concurrent model development and analysis. The tool has built-in classifiers by default or can train a model in real time given the data. The interface is user friendly and the results can be displayed in a tabular format or on an interactive map. The tool also provides an option for users to download the word with their probability scores and the results in csv files.

The advantages and limitations of each proposed methodology were discussed, and several future research directions were outlined. In summary, the methodologies and tools developed as part of the dissertation can assist transportation engineers and safety professionals in extracting valuable information from narratives, recovering missed crashes, classifying a new crash, and expediting their review process on a large scale. Thus, this research can be used by transportation agencies to analyze crash records, identify appropriate safety solutions, and inform policy making to improve highway safety of our transportation system.

Keywords: Crash narrative; Text analytics; Natural language processing; Machine learning; NoisyOR; Attention Layers; Work zone, Inattentive Driving; Distracted Driving; Crash Sequence; Crash Events; Web tool; Traffic safety

© Copyright by Md Abu Sayed, 2022  
All Rights Reserved

## TABLE OF CONTENTS

Table of Contents .....	vi
List of Figures .....	xii
List of Tables .....	xiv
ACKNOWLEDGEMENTS .....	xv
Chapter 1. INTRODUCTION .....	1
1.1    Background .....	1
1.2    Problem Statement .....	4
1.3    Research Objectives .....	7
1.4    Dissertation Outline.....	9
Chapter 2. LITERATURE REVIEW .....	12
2.1    ML and NLP Techniques in Text Analytics .....	12
2.2    Text Analytics in Transportation Safety Research.....	14
2.2.1    Application of Traditional Text Analytics .....	14
2.2.2    Application of Machine Learning .....	17
2.2.3    Application of Deep Neural Network (DNN).....	18
2.2.3.1 <i>Attention Layer in Transportation Research</i> .....	20
2.2.3.2 <i>Attention Layer in Text Analytics</i> .....	22
2.3    Critical Findings in Existing Studies.....	24
Chapter 3. RECOVERING MISSED CRASHES FROM CRASH NARRATIVES- A NOISYOR METHOD.....	30

3.1	Introduction .....	30
3.2	NoisyOR-based Classification .....	32
3.2.1	Equation of Simple Count Probability .....	33
3.2.2	Equation of Weighted Count Probability.....	35
3.3	Case Study 1: Work Zone Crashes.....	36
3.3.1	Data Collection .....	37
3.3.1.1	<i>The Nature of Noisy Data</i> .....	39
3.3.1.2	<i>Data Cleaning and Pre-processing</i> .....	40
3.3.2	Crash Verification and Model Performance Evaluation .....	41
3.3.3	Results and Analysis.....	41
3.3.3.1	<i>The Analysis of Positive Unigram and Positive Bigram</i> .....	42
3.3.3.2	<i>Comparing Unigram and Unigram +Bigram Methods</i> .....	45
3.3.3.3	<i>Extended Analysis of Unigram+Bigram of NoisyOR</i> .....	46
3.3.3.4	<i>Analysis of missed WZ crashes</i> .....	49
3.3.4	Summary of Observations.....	52
3.4	Case Study 2: Distracted and Inattentive Driving Crashes .....	54
3.4.1	Data Collection .....	56
3.4.2	Result and Analysis.....	59
3.4.2.1	<i>Unigram, Bigram, and Trigram Probability Analysis</i> .....	59
3.4.2.2	<i>DOI, DD, and ID Classifiers</i> .....	64
3.4.3	Summary of Observations.....	69
3.5	Conclusion and Recommendations .....	71
	Chapter 4. METHODOLOGY COMPARISON .....	75

4.1	Introduction .....	75
4.2	Methodologies .....	76
4.2.1	Multinomial Naive Bayes (MNB) .....	77
4.2.2	Logistic Regression (LGR) .....	77
4.2.3	Support Vector Machine (SVM).....	78
4.2.4	Random Forest (RF) .....	78
4.2.5	K-Nearest Neighbor (K-NN) .....	79
4.2.6	Gated Recurrent Unit (GRU) .....	80
4.3	Data Collection and Pre-Processing.....	80
4.4	Feature Generation and Model Parameters Tuning.....	81
4.5	Evaluation Methodologies.....	83
4.5.1	Evaluation of Classification Performance .....	83
4.5.2	Evaluation for Identifying Missed WZ Crashes .....	84
4.6	Findings of WZ Related Words .....	84
4.7	Model Evaluation and Discussion.....	88
4.7.1	Analysis of Overlapping Cases for LGR, SVM, NoisyOR and GRU .....	89
4.7.2	Comparison between GRU and NoisyOR .....	92
4.8	Summary of Observations.....	95
4.9	Conclusion and Recommendations .....	97
Chapter 5.	CLASSIFYING MOTOR VEHICLE CRASHES USING SENTENCE-BASED HIERARCHICAL ATTENTION NETWORKS .....	100
5.1	Introduction .....	100
5.2	Crash Data Collection and Preprocessing .....	102

5.3	Sentence-based Hierarchical Attention Network (SHAN).....	106
5.3.1	Word Level Attention .....	108
5.3.2	Sentence Level Attention.....	110
5.3.3	Classification.....	111
5.4	Data Preprocessing and Model Parameter Setup .....	112
5.5	Result and Discussion .....	114
5.6	Conclusion and Future Works.....	119
Chapter 6. DESIGN AND DEVELOPMENT OF TEXT INTELLIGENCE SOLUTION TO IMPROVE HIGHWAY SAFETY.....		121
6.1	Introduction .....	121
6.2	Crash Narrative Analysis: Issues and Challenges .....	123
6.3	Algorithm .....	125
6.3.1	Step 1 Create Event-related Keywords .....	127
6.3.2	Step 2 Create Action Keywords.....	128
6.3.3	Step 3 Identify Contents by Unit Numbers.....	128
6.3.4	Step 4 Create Sequence of Event by Unit Numbers .....	129
6.4	Result Validation.....	132
6.5	Application of the Results .....	134
6.5.1	Finding Additional Events in the Crash Narrative.....	135
6.5.2	Identify the Scenarios (Subsequence) in (K+A) and (B+C) Injury Crashes.....	139
6.5.3	Spatial-temporal Analysis of the Sequence of Crash Events.....	146
6.6	Conclusion.....	148
Chapter 7. DEVELOP A CRASH INFORMATION EXTRACTION, ANALYSIS AND CLASSIFICATION TOOL (CIEACT) USING TEXT MINING TECHNIQUES .....		151

7.1	Introduction .....	151
7.2	Functionalities .....	152
7.3	System Architecture .....	154
7.3.1	Backend.....	156
7.3.1.1	<i>Web Framework</i> .....	156
7.3.1.2	<i>Database</i> .....	161
7.3.1.3	<i>Classification Algorithm</i> .....	161
7.3.2	Frontend .....	164
7.3.2.1	<i>Step (page) 1</i> .....	164
7.3.2.2	<i>7.3.3.2 Step (page) 2: “Default Model” and “Train Model” Page</i> .....	165
7.3.2.3	<i>Step (page) 3: Result Summary and parameter selection page of “Train Model”</i> .....	166
7.3.2.4	<i>Step (page) 4 of Data upload Page for “Train Model”</i> .....	166
7.3.2.5	<i>Step (page) 3 of “Default Model” and Step 5 of “Train Model”</i> .....	168
7.4	Evaluation and Testing.....	170
7.4.1	Evaluation of the Output of the Tool .....	170
7.4.2	Browser Compatibility, HTML and CSS Syntax Validity Testing .....	171
7.5	Case Study .....	171
7.5.1	Case Study with Small Dataset .....	172
7.5.2	Case Study with Large Dataset .....	174
7.6	Conclusions and Recommendations.....	176
Chapter 8.	CONCLUSION AND FUTURE RECOMMENDATIONS .....	178
REFERENCES	.....	184

APPENDIX A: Pseudocode for PT algorithms .....	204
APPENDIX B: Pseudocode for PMPT algorithms.....	205
APPENDIX C: POS tags of PMPT methods.....	207
APPENDIX D: Pseudocode for DP method.....	209
APPENDIX E: The list of candidate words and phrases extracted from narratives.....	210
APPENDIX F: Unigrams for WZ NoisyOR Classifier .....	211
APPENDIX G: Top 20 Unigrams and Bigrams from Model Training .....	212
APPENDIX H: Trained Model Summary .....	213
APPENDIX I: Top 100 Unigrams and Bigrams from Model Training .....	215
APPENDIX J: Train Model Statistics.....	217
APPENDIX K: Homepage (Top) and About page (Bottom) of CIEACT.....	219
APPENDIX L: Contact page (Top) and Help page (Bottom) of CIEACT.....	220

## LIST OF FIGURES

Figure 2.3-1 Research Objectives vs Methods (Top) and Research Objectives Vs Data (Bottom) .....	27
Figure 3.3-1 Accuracy of (Unigram+Bigram) NoisyOR .....	47
Figure 3.3-2 Histogram of Narrative Length for a) NWZ and b) Missed WZ .....	48
Figure 3.3-3 WZ Crash Analysis by a) Hour b) Day and c) Month .....	50
Figure 3.3-4 WZ Crash Analysis by Highway Class .....	51
Figure 3.4-1 DOI Classifier Using the U+B Approach (Eq. 3, 0.50 Cutoff Threshold).....	65
Figure 3.4-2 DD Classifier Using the U+B Approach (Eq. 3, 0.65 Cutoff Threshold).....	66
Figure 3.4-3 ID Classifier Using the U+B Approach (Eq. 3, 0.75 Cutoff Threshold) .....	67
Figure 4.6-1 Important Words Found by LGR and SVM.....	86
Figure 4.6-2 Top 15 Positive Unigrams and Bigrams Obtained by the NoisyOR.....	87
Figure 4.7-1 Missed WZ Crash Detection Accuracy (%).....	89
Figure 4.7-2 Distribution of Narrative Length.....	92
Figure 4.7-3 Accuracy of Noisy-OR and GRU .....	94
Figure 5.2-1 Methodology for Data Balancing.....	104
Figure 5.2-2 Optimal Topic Number Selection .....	105
Figure 5.3-1 Hierarchical Attention Model (HAN) (Z. Yang et al., 2016).....	106
Figure 5.3-2 SHAN Architecture for Classification .....	108
Figure 5.5-1 Attention Weights of Words in Various Contexts .....	117
Figure 5.5-2 Words with High Attention Scores and Their Relative Importance .....	118
Figure 6.3-1 Pseudocode of HGEN Algorithm.....	131
Figure 6.4-1 Findings from Manual Evaluation .....	133

Figure 6.5-1 Percentage of Additional Events by Injury Severity - (a) Regardless the Location of Most Harmful Events, (b) Only before the Most Harmful Events .....	136
Figure 6.5-2 Most Frequent Events Associated with Missed ROR in Structure Data.....	137
Figure 6.5-3 Additional Events found before the Most Harmful Events in Crash Narratives....	138
Figure 6.5-4 Most Frequent Subsequences in K+A and B+C Crash Sequences .....	140
Figure 6.5-5 Association of Most Harmful Event with the Subsequence of Events Occurred before the Most Harmful Events (“e” in parenthesis) in K+A.....	143
Figure 6.5-6 Association of Most Harmful Event with the Subsequence of Events Occurred before the Most Harmful Events (“e” in parenthesis) in B+C.....	145
Figure 6.5-7 Approximate Jurisdictional Location of Crash Report Officers and the Rate of Crash Report Contained Additional Events .....	148
Figure 7.3-1 System Architecture (Deremuk, 2021) .....	155
Figure 7.3-2 Overview of the Web Framework for CIEACT.....	158
Figure 7.3-3 Logical Diagram of Functionalities of CIEACT .....	159
Figure 7.3-4 SQL Script to Upload the Word Probability csv Data File (to be used with the Default Model).....	161
Figure 7.3-5 Overview of the Frontend of the CIEACT.....	164
Figure 7.3-6 The functionalities of Default Model (left) and Train Model(right).....	165
Figure 7.3-7 Screenshot Showing the Result Summary Page of CIEACT .....	167
Figure 7.3-8 Screenshot Showing the Result Visualization Page of CIEACT .....	169
Figure 7.5-1 Number of Secondary Crashes vs. Cut-off Value of the Classification Score.....	173
Figure 7.5-2 Secondary Crashes vs. the Cut-off Value of the Classification .....	175

## LIST OF TABLES

Table 2.3-1 Summary of Literature Review .....	26
Table 3.3-1 Top Ten Positive Unigrams and Bigrams by Probability Score Using Equation 1... ..	43
Table 3.3-2 Top 15 Positive Unigrams and Positive Bigrams By df, cf, and Probability Score ..	43
Table 3.4-1 Crash Statistics .....	58
Table 3.4-2 Top 25 Unigrams, Bigrams and Trigrams for the DOI Classifier .....	60
Table 3.4-3 Top 25 Unigrams, Bigrams and Trigrams for the DD Classifier .....	62
Table 3.4-4 Top 25 Unigrams, Bigrams, and Trigrams for the ID Classifier .....	63
Table 3.4-5 Summary of All Results of Best Approaches .....	68
Table 4.7-1 Overlapping True WZ among LGR, NoisyOR, SVM and GRU. ....	90
Table 5.4-1 Model Evaluation for Inattentive and Distracted Crash Classification .....	113
Table 5.5-1 Example of Most Weighted Sentence and Words Visualization for Distracted (DD) and Inattentive Driving (ID) Crashes.....	116
Table 6.5-1 Variables Associated with Additional Events .....	146
Table 7.3-1 Descriptions of the Views and Algorithm in the Logical Diagram .....	160

## **ACKNOWLEDGEMENTS**

First and foremost, I would like to express my heartfelt gratitude to Almighty Allah for putting me on this wonderful, eventual, and memorable journey and for providing me with the courage, knowledge, patience, strength, and confidence to overcome all the challenges on the way to completing my PhD. The individual who has made things possible for me and who has always guided me along this path is my advisor professor Xiao Qin, a very exceptional person.

Professor Xia Qin is well-known for being an excellent mentor, advisor, and researcher. But, to me, he is more than that, who discovered me, my strength and weakness and molded me to a new form. His mentorship made me courageous, confident, and curious to tackle each obstacle in my path. I would like to thank him for teaching me how to walk in the traffic safety research.

I would like to thank my committee members, Dr. Rohit J Kate, Dr. Habib Tabatabai, Dr. Susan W Mcroy, and Dr. Yin Wang, for serving on my committee, devoting their time to reading my research proposal and dissertation draft, and providing me with insightful comments and suggestions. I would like to thank all my colleagues for their continuous support. I would like to thank Wisconsin Department of Transportation (WisDOT) for providing me with data.

I would like to thank my wife, Mitu, who is a real fighter, for her patience, love, and support, as well as my daughter, Shifa, who never fails to make me smile. I would also like to thank the entire Bengali community of Milwaukee, especially Manzur vai and Saber vai, and their families.

# **Chapter 1. INTRODUCTION**

## **1.1 Background**

Safety practitioners are ceaselessly working to ensure that travelers arrive at their destination without incident. Numerous safety agencies have the goal of achieving zero fatalities on the road, which entails a completely safe road environment. However, not every trip is incident-free. In reality, the number of fatalities and injuries resulting from traffic accidents remains a significant public health concern. According to the World Health Organization (WHO), more than one million people, including drivers, passengers, pedestrians, bicyclists, and transit users, are killed and between 20 and 50 million suffers non-fatal injuries from road accidents each year around the world (WHO, 2022). Despite the significant advances in vehicle technologies, safety data collection and analysis, and engineering improvements, over 30,000 people die (NHTSA, 2020) in road crashes each year in the United States of America. What is alarming is that the trend of fatal and serious injury crashes seems to move in the wrong direction. Twelve-year statistic from 2011 to 2021 shows that the fatality rate has been increased each year. The actual rate of fatalities in 2021 exceeds the projected rate. This challenge calls for a more in-depth and advanced analysis to determine how to improve safety by determining the causes of the crash (esp. fatal and major injuries) and the ways to prevent them from happening. It requires holistic approaches, such as analyzing problems from multiple perspectives, utilizing available data sources, and employing the most appropriate tools and technologies from within and outside the domain.

The crashes are complicated because they are not only caused by a multitude of factors, but also the result of a series of actions and events. Although standard safety practice is to identify and eliminate crash risk factors, breaking the chain of events or actions could be another way to prevent a crash from reaching its most catastrophic stage. There are opportunities for safety intervention based on the nature and order of events preceding the most hazardous events, many of which have been recorded in the crash report, either in structured data fields or unstructured crash narratives.

Numerous studies have shown that structure crash data (also called tabular data) is insufficient because many crashes are missed or misclassified, which in turn causes one crash to be underestimated while the other is overestimated. (Elvik & Mysen, 1999; E Hauer & Hakkert, 1988; Sayed et al., 2021; Tsui et al., 2009; Ye & Lord, 2011a). Crashes are missed in the structure data field for a variety of reasons: restrictive reporting options in tabular forms (Blackman et al., 2020; Ullman & Scriba, 2004; J. Wang et al., 1996); lack of understanding about the importance of the crashes, overloaded by work during crash reporting time (Graham & Migletz, 1983); and misclassification of crashes (Wang et al., 1996, Farmer, 2003). Generally, a police officer makes certain judgments about a crash based on the severity of the crash and the driver. A fatal crash is usually given the highest reporting priority, compared with property damage crashes which usually receive a lower priority (Ye & Lord, 2011b). A crash with less severity or no injuries are usually not reported in structure data (J. Wang et al., 1996). In addition, the probability of reporting an injury crash increases with the age of the injured person (i.e., for young children, it is 20-30%; and for persons over 60, it is 70%); and the number of vehicles involved (Ezra Hauer & Hakkert, 1988). A crash involving a younger or female driver

has a lower probability of being reported (Amoros et al., 2006). Therefore, estimates based solely on structure data fields underestimate the results of the safety analysis (Abay, 2015).

As a supplement to structure data, many professionals in the field of traffic safety rely on crash narratives because crash narratives not only fill in information gaps but also reveal previously unknown facts about crashes. The narrative includes additional details about all the vehicles involved in the accident, including any citations, witnesses, drugs or medication, hazardous material spills from trucks and buses, trailer and towed vehicle information, school bus details, etc. Information in the crash narrative is very important because it describes the unique and different circumstances of each crash scene. The information is especially helpful when looking into crashes that were misclassified or were missed.

Crash narratives are very similar to the text data that Natural Language Processing (NLP) researchers often use in other fields (e.g., grammatical structures, information flows); but, narratives have their own challenging features such as:

- 1) *Irrelevant information (noise)*. In the majority of the narratives, sentences contain information that is unnecessary for safety analysis.
- 2) *Missing relevant information (noise)*. Many crash narratives lack information pertinent to the crash for which they are categorized.
- 3) *Highly imbalance data*. The classification problem becomes one versus all as a result of the large variety of different crash types, which ultimately leads to an excessively unbalanced dataset.

- 4) *Varying narrative length.* The length of the narratives varies from just a few hundred to over two thousand characters.

Some of these issues can be handled by the methods and techniques in the domain of text analytics. The NLP and Machine learning (ML) are the most used techniques in text analytics to automate the processing of unstructured text. NLP is an automated text data analysis technology that employs a set of theories and tools to simulate human language processing capabilities. (Liddy, 2001). An important branch of NLP is text classification that seeks to label texts based on their contents or contexts. Text classification has recently received significant interest due to its usage in e-mail filtering, spam detection, web-page content screening, automatic message routing, automated article indexing, and searching for relevant material on the Web (Kowsari et al., 2019). Another major aspect of NLP is pattern mining, which identifies hidden knowledge in unstructured text data irrespective of how the information is presented in the text. For decades, a number of ML approaches with NLP have been developed and successfully employed for pattern identification and text classification. The availability of textual datasets and the growing interest in NLP and ML techniques have incited research in transportation engineering, especially in the analysis of highway safety data.

## 1.2 Problem Statement

Traffic safety research has been hampered by several factors, such as a lack of focus on the available data, methods, and tools. In addition to traditional data, additional data sources can provide extra and valuable information for untangling a complex problem that may not be

solvable using traditional data alone. Alternative data can also be used to check the validity of the traditional data, thereby improving data quality. In addition, it provides the opportunity to examine a problem from different perspectives. However, dealing with nontraditional data presents several unanticipated challenges in determining the appropriate methods and tools to process the data, generate and interpret the results. Therefore, the problems in traffic safety analysis are many folds, as discussed below:

- *Stagnant safety statistics in fatal and seriously injured crashes.* Despite significant improvements in vehicle technologies, safety data collection and analysis in the transportation field, the number of injuries and fatalities has remained constant over the decades. It necessitates holistic approaches, such as analyzing problems from multiple perspectives, utilizing available data sources, and employing the most suitable tools and technologies from within and outside the domain.
- *Under-utilization of rich information in crash narratives.* Even though crash narratives contain a lot of information that are missed in structured data, including new information, the safety practitioners review the narratives manually, which is labor intensive, time consuming and the review quality is inconsistent. Law enforcement officers use different words or phrases in the narratives, which presents a challenge for traffic safety engineers when querying specific terms. Because of these facts, practitioners can only review a small number of narratives to serve their specific purpose. As a result, much knowledge remains unexplored. On top of that, review quality can vary widely depending on the expertise and opinion of the people conducting the review. Therefore, research is needed to automate crash narrative analysis, expand its scalability, and ensure reasonable

accuracy. A comprehensive analysis of crash narratives utilizing text analytics can provide valuable insights about the use of crash narratives.

- *Lagging of DM/ML advances in safety analytics.* Despite attention layer-based DNNs with NLP have shown success in transportation-related forecasting, prediction, object identification, as well as text classification problems, it has never been applied to safety analysis using textual data. Attention layers are good at capturing the contexts of words, mitigating the consequences of incorrect or missed labeling and interpreting the result.
- *Lack of intelligent method for extracting crash events in sequence.* A crash consists of a series of events and the actions of drivers. Breaking the sequence of events or actions could be an additional method of preventing a crash from reaching its most catastrophic phase. The more events that happen before the most harmful event of a crash, the more chances there are to stop the crash. The crash narrative supplements structure data events with additional and missing information. However, there is no algorithm that can automatically extract events from crash narratives. The manual efforts to extract crash events from the narratives and the frequency of missing information in the structure data impeded progress in safety research.
- *Limited tools available to practitioners.* In the existing literature, text mining and machine learning techniques have been shown to be an efficient and effective way to automatically extract important information from crash narratives. However, the benefits of these studies are not readily available because there is no tool that allows safety professionals to use these techniques for crash narrative analysis.

Solving these problems is critical for advancing traffic safety research. Particularly, it is evident that a deep investigation into both structured data and crash narratives using text analytics is necessary to diagnose the problems. In addition, sophisticated tools and a well-designed methodology are required to manage and investigate large crash reports. These tools and techniques can assist in achieving several research objectives, such as the classification and recovery of missed crashes, the identification of crash factors and crash patterns, and the generation of crash sequences from crash narratives.

### **1.3 Research Objectives**

The goal of this dissertation is to advance the use of machine learning (ML) and natural language processing (NLP) in rapid and efficient analysis and retrieval of underutilized and unexplored critical information from police crash narratives. The algorithms and methods developed from the dissertation will improve and optimize analyses using text data and support data-driven safety analysis tools. This goal will be achieved through the following objectives:

- 1) Investigate state-of-the-art text mining techniques in transportation safety research.
- 2) Conduct an in-depth analysis of crash narratives, discuss its utility over structure data, and discuss the major challenges extracting information using text analytics.
- 3) Develop crash classification algorithms to recover missing crashes from crash narratives.

- 4) Conduct a comparative analysis for recovering missed crashes using state-of the art text analytics and machine learning techniques used in transportation safety and discuss their utility and limitations in conducting traffic safety research.
- 5) Discuss the benefits of using text analytics with advance deep learning in transportation safety analysis.
- 6) Develop crash sequence generation algorithms to automate crash event extraction from crash narratives.
- 7) Discover patterns in the sequence of crash events, especially in fatal and injury crashes, and discuss their potential application in traffic safety research.
- 8) Develop a text analytic web tool using NLP and Geographic Information System (GIS) to investigate and analyze crashes using crash narratives.

This dissertation will detail text analytics suitable for traffic crash analysis, the benefit of the developed algorithms, and the findings of the research outcomes. Such information is critical for determining the factors that contribute to crashes and comprehending the crash scenarios. Furthermore, emphasizing the significance of the crash narrative encourages and incentivizes law enforcement agencies to use narratives to capture data that is not available in data fields.

## **1.4 Dissertation Outline**

The dissertation is divided into eight chapters based on the proposed research objectives.

The dissertation is structured as follows:

Chapter 2 provided a comprehensive review of existing literatures conducted for traffic safety research using text analytics and machine learning, including possible deep learning research used for structure data that could be considered for text analytic applications. The focus was on crash narrative studies to investigate methodological alternatives such as machine learning and natural language processing-based techniques to address safety-related critical issues; research topics; characteristics of text data and critical data issues; challenges in crash mining; and the value of crash narratives in vehicle safety research. This chapter provided a summary of the benefits and limitations of these studies, as well as a list of important research questions. The investigation into the relationship between research objectives, methods, and data revealed research gaps and provided valuable insights into text analytics in traffic safety in terms of data sources, methods, and their relationship to other data and research objectives.

Chapter 3 presented NoisyOR algorithms for recovering missed crashes from narratives. Two case studies, Work-zone crashes (WZ) and Distracted crashes (DD) and Inattentive crashes (ID), were used to discuss the performance of the algorithm and analyze the results. The algorithms extract crucial crash information from narratives while classifying narratives.

Chapter 4 compared multinomial naive bayes (MNB), logistic regression (LGR), support vector machine (SVM), k-nearest neighbor (K-NN), random forest (RF), gated recurrent unit (GRU), and NoisyOR for classifying crashes using crash narratives. It examined the advantages

and disadvantages of each technique for classifying missed crashes and extracting relevant data from crash narratives. This study examined narratives in depth and identified several issues that could be used to guide model selection when working with crash narratives.

Chapter 5 developed a novel neural network architecture named Important Sentence-based Hierarchical Attention Networks (SHAN) to classify crashes from narratives. Text analytics have been discussed in relation to the utility of attention-based neural networks. It described the architecture of the proposed SHAN and how it has been improved in comparison to existing models. The model utilized a word and sentence level attention layer with a GRU neural network to generate an attention score for each sentence and word, and then used the sentence with the highest attention score for classification in the second step. The performance of SHAN was compared to that of Gated Recurrent Unit (GRU) and Hierarchical Attention Network (HAN) in classifying crashes based on crash narratives. The LDA method was utilized for the first time to address data imbalance issues, and its efficacy has been analyzed. The interpretability of the result through visualization provided valuable insight.

Chapter 6 presented the development of four NLP and ML-based algorithms to generate sequence of crash events form crash narratives. The strengths and weaknesses of each algorithm were outlined, and the best algorithm was analyzed in depth. The results were validated based on the temporal order of events in the sequence for the entire dataset, as well as the manually reviewed outcomes to validate the results. This research explained why the proposed algorithms are necessary to complement structured data. The results were discussed in terms of fatal and injury crashes and the most hazardous incidents.

Chapter 7 developed Crash Information Extraction, Analysis and Classification Tool (CIEACT), an online web software, to analyze crash narratives using text mining techniques. The tool provides a default model and a model that can be trained on the fly to classify crashes. Both the default model and model training use Noisy-OR. The tool has been tested offline and online to ensure device compatibility. The tool's user interface has navigation instructions. The tool displays results with color coding and classification score in tabular form and offers a map option. Finally, users can download word probability and results in csv files for analysis.

Chapter 8 summarizes the research conducted in this dissertation and its contributions. The limitations and future directions of the research are also presented.

## **Chapter 2. LITERATURE REVIEW**

The use of text analytics is comparatively new and slow in transportation domain with respect to other domains. A thorough analysis of the literature review can reveal the breadth of existing text analytics research in terms of methodology, data, and research objectives. Therefore, this chapter intended to capture a wide range of existing transportation research that were basically conducted for transportation text data analysis. In particular, this chapter provided a comprehensive review of previous and current text analytics research in the transportation field, including (1) various sources of text data for traffic safety analysis, (2) techniques for addressing critical challenges in textual data, (3) text mining topics in traffic safety, (4) and the comparison and evaluation of various text mining methodologies. A comprehensive review of crash narrative studies covered a wide range of topics, including (1) crash classification, (2) crash contributing factors, (3) crash scenarios, (4) and the extraction of new information from crash narratives. The attributes of various methods, the logic of any underlying assumptions, methodological and data limitations, data requirements, model performance, and future research directions are discussed.

### **2.1 ML and NLP Techniques in Text Analytics**

ML and NLP techniques are the most widely used techniques in text analytics to analyze unstructured text data. Text analytics ( also known as text mining) is the automated extraction of

previously unknown information from text and the linking of the extracted information to form new facts or hypotheses to be explored further through more traditional means of experimentation (Hearst, 2003). NLP is a set of theories and technologies designed to achieve human-like language processing (Liddy, 2001). Text mining uses NLP to process data and extract valuable insights from it. ML techniques are mostly being used for prediction, forecasting and classification based on hidden information in the text.

Text analytic has become popular and necessary in many fields, including financial services, health care, transportation, communication and media, information technology and internet, political analysis, public administration and legal services (Gupta et al., 2009; Inzalkar & Sharma, 2015; Maheswari & Sathiaseelan, 2017). Information retrieval, natural language processing, information extraction, text summarization, opinion mining and sentiment analysis are some of important areas of text mining research (Allahyari et al., 2017).

In text analytics, the problems are solved by classification, prediction, clustering, or information extraction techniques. Some commonly used clustering algorithms are hierarchical clustering, k-means clustering, and probabilistic clustering and topic models (e.g., probabilistic latent semantic analysis, latent Dirichlet allocation) (Allahyari et al., 2017). Examples of popular classification algorithms include naive bayes, nearest neighbor, decision tree, decision rule, support vector machine, logistic regression, Rocchio's algorithm, associative classifier, centroid based classifier, and neural network (Allahyari et al., 2017; Brindha et al., 2016; Korde & Mahender, 2012). Besides, various deep learning algorithms such as Convolutional Neural network (CNN), Recurrent Neural Networks (RNN) are noteworthy.

## **2.2 Text Analytics in Transportation Safety Research**

The use of text analytics in the transportation field is diversified. For example, the diagnosis of vehicle on-board equipment faults (Zhao et al., 2014), recognition of license plate (Oliveira-Neto et al., 2009), detection of behavioral failure in electric vehicle infrastructure (Ha et al., 2020) and user satisfaction surveys (Qi et al., 2020; Serna & Gasparovic, 2018). In traffic safety, the notable research of text mining relates crash severity analysis, crash contributing factor identification, crash event detection, crash propagation and cause analysis, and crash type classification. To further discuss the use of text analytics in transportation domain, this section has been divided into the following three subsections:

### **2.2.1 Application of Traditional Text Analytics**

Numerous research has been conducted to solve a wide variety of transportation-related problems through information extraction from text data. Such as, Sorock et al. (1996) used the keyword “construction” to classify the narratives as work zone crashes. They analyzed insurance claim narratives for five pre-crash activities and crash types in work zones. The pre-crash activities are stop, cut, driver error, back, and merge, and the five crash types are hit objects(small), hit object (large), flipped, side impact, and rear end. The analysis was conducted with the help of a statistical analysis system (SAS) that selected keywords. The frequency of the words was used to determine the causative factors of the crashes. Then, the similar keywords were grouped and classified as either one of five pre-crash activities or one of five crash types. However, a work zone can be identified by a number of different keywords, such as

"construction zone", "due to construction", "construction barrel", and so on. Therefore, the method based on only keyword can provide many false positives and false negatives.

Nayak et al. (2009) generated cluster of topics by applying Bayesian theory based Leximancer tool to find the major contributing factors of crashes from crash narratives. According to Leximancer white paper (Leximancer, 2010), Leximancer extends and reworks the techniques of Latent Semantic Analysis (LSA), Hyperspace Analog to Language (HAL), and Latent Dirichlet Allocation (LDA). It incorporated two stages of non-linear machine learning to provide a statistical method for extracting semantic patterns from text. However, detailed information about the methods is not publicly available, preventing inferences regarding the extent of improvement.

Gao & Wu (2013) developed a verb-based text mining method by applying various NLP techniques that automatically identify action words (verbs) from crash narratives. The method utilized syntactic and semantic information from the text to overcome the limitations of their previous methods that used predefined keywords. However, the process was not completely automatic, as the words with similar meaning had to be grouped together manually.

Roberts & Lee (2014) investigated the driver distraction based on word frequency and association rules using Twitter data. Their associations rules revealed useful insights into major crash factors and their associations with other words.

D. Zheng et al. (2015) used a three-step procedure involving a spatial-temporal window, two filters, and manual validation for secondary crash analysis. They empirically determined the spatial and temporal boundaries of the spatial-temporal window. As filters, they identified crash

falls in impact zones such as queues and congestion areas. The result was then manually verified. They proposed an automatic secondary crash verification process that utilizes keywords and the distance of keyword. The distance was calculated by the absolute difference in indexes between two types of keywords: relationships keywords (RKWs) and events keywords (EKWs).

Rakotonirainy et al. (2015) used SAS Text Miner 3.1 and Ward algorithm that provides a clustering mechanism based on frequency of keywords to identify crash contributing factors. They applied rough set analysis that identified minimal subset of crash contributing factors and their interrelationships by discovering hidden pattern in the data. The words mentioned only in curve-related crashes were selected as keywords, and the keywords with high frequencies were used as the main factors contributing to curve-related crashes.

Williams & Betak (2018) conducted a comparative study between latent semantic analysis (LSA) and latent dirichlet allocation (LDA) text mining techniques, to detect different types of crashes based on the topics from narratives. They found both methods complement each other to categorize crashes.

Maghrebi et al. (2015) analyzed location-activity patterns applying cluster text mining techniques using Twitter data. Brown (2016) discovered the predictive factors or contributors that lead to rail accidents by mining the text narratives of rail accident reports. (Park et al. (2018) developed association rules to analyze the association between words related to the type of work being performed and the type of lane closure in expressway work zones. They used textual data that contained work zone information on the expressways. Trueblood et al. (2019) developed a classifier tool in Excel to identify agricultural crashes from crash narratives. The authors prepared two lists of keywords (agricultural and nonagricultural) manually and used the lists to

search keywords in the narratives for identifying the agricultural crashes. However, their classifier assigns equal weight to the narratives that are related to agricultural crash, so it may not be effective for large data sets in which narratives are more relevant to agricultural crash. Kwayu et al. (2021) investigated crash factors and co-occurrence of the factors that contribute to crash incidents by mining text narrative.

## **2.2.2 Application of Machine Learning**

Meanwhile, a variety of machine learning techniques, such as Bayesian classifiers, support vector machines, k nearest neighbors, decision trees, and neural networks have been successfully used for text classification for decades (Y. Yang, 1999). For example, J. Zhang et al. (2016) conducted a comparative study on naive bayes (NB), support vector machine (SVM) and decision tree (DT) methods to find hazardous behaviors of drivers from the crash narratives and found that NB is the best binary classifier. However, the process is not fully automatic because narratives are manually annotated for training and testing the model. Another study conducted by Proof (2019) to classify pedestrian crashes from text narratives also relied on manual review to prepare training dataset.

Fitzpatrick et al. (2017) used logistic regression (LG) to identify speed-related missed crash from noisy crash reports. Their investigation showed that almost half (46.6%) of the crash narratives did not have any information about speed related crashes. Their model was successful identifying speed related crashes from narrative that contained speed related crash factors.

Das et al. (2020) used SVM, RF and XGBoost techniques to classify pedestrian crashes from text narratives and found that XGBoost has an accuracy rate of 72%, which is more accurate than the other two models. However, they used a very small dataset to train and test the model, which may weaken the validity of the results. In a similar study but for different crash types such as tree and utility pole related injury crashes, found similar performance for XGBoost. They used LIME with the XGBoost to extract the important words from the result (Das, Datta, et al., 2021). Arteaga et al. (2020) used a wide variety of machine learning techniques that included NB, SVM, LR, XGBoost, random forest (RF) and neural network (NN), to classify hospitalization vs fatal crash from narratives and found NN provided best performance results. The crash factors were extracted from NN results using Lime. The advantage of using LIME is that it does not consider linearity in parameters that is suitable for NN and also provides interpretability of the results.

McAdams et al. (2018) used multivariate logistic regression to study the role of helmets in reducing the injury rate of bi-vehicles using narratives collected from the national electronic injury surveillance. The narrative describes the events of actions that occurred at the time of accident, and these events help determine the importance of helmet use.

### **2.2.3 Application of Deep Neural Network (DNN)**

The previous section demonstrated that text mining techniques based on NLP and machine learning techniques (which are mostly known as shallow machine learning), such as, NB, SVM, DT, XGBoost, DT, and RF, produced valuable results in terms of information

extraction and classification problems. However, the disadvantage of using these methods is that the results are generated based on frequency of the words, without considering the context of the words. In that case, a classifier will misclassify a narrative when a word is used in a different context in the text. Word embedding is the most prevalent technique used in deep neural network (DNN) to incorporate words with their contextual meanings. It transforms a word into multidimensional word vector while preserving its semantic meaning. Even though the use DNN in text analytics has advanced significantly in other fields, it has not been widely explored in the transportation field.

Heidarysafa et al. (2019) used various combinations of deep learning models to study the use of text narratives in determining the causes of railway accident. In order to check whether the reported crash causes are consistent with the narrative description, they used various combinations of one-dimensional Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) . The RNN includes Long Short Term Memory (LSTM) and Gated Recurrent Unit (GRU) and the word embedding includes word2vec<sup>1</sup> and GLoVe <sup>2</sup>. Although CNN and RNN with word2vec provided better classification results compared to other models, these models did not work well for minority classes in the dataset.

---

<sup>1</sup> word2vec is not a singular algorithm, rather, it is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets. Embeddings learned through word2vec have proven to be successful on a variety of downstream natural language processing tasks.

<https://www.tensorflow.org/tutorials/text/word2vec>

<sup>2</sup> GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. <https://nlp.stanford.edu/projects/glove/>

Undoubtedly, there is a paucity of text analytics research in traffic safety research using deep learning. In other text analytics research domains, the DNN is utilized to improve the application of natural language processing by automatically learning the context of the data, thereby eliminating the need for human intervention. Among many intelligent methods, NLP with attention layer based RNN has recently piqued the interest of other researchers. Although RNN (e.g., LSTM, GRU) can capture word contexts, it is ineffective in longer text. The attention layer, as opposed to traditional RNN, can detect, and learn the important parts of the text more efficiently and effectively from both shorter and longer narratives. It deemphasizes the less important parts of the text (such as noisy words and sentences). In other words, the attention layer can learn neighboring data and reveal the relationship between them. It is a widely used technique for mitigating the effects of mislabeled data (Lin et al., 2016; Yin et al., 2022; Zhu et al., 2019).

#### *2.2.3.1 Attention Layer in Transportation Research*

The use of attention layer in transportation research is evident but limited. The majority of research employed it for prediction purposes such as traffic flow, routing, multi modal demand, vehicle movement, travel time and delay prediction, yielding some valuable insights into the attention mechanism. For example, Y. Wu et al. (2018) applied attention layers with CNN and RNN, in which attention layer helped to capture the short-term traffic flow pattern and the relation with flow and speed. Attention layer can be used multiple times in different part of the model to serve different purposes. Z. Zhang et al. (2019) combined a graph convolutional

network and an attention mechanism within a Seq2Seq<sup>3</sup> framework to create a prediction model capable of depicting spatial-temporal correlations in multistep traffic prediction. Their results indicate that the attention layer produces high attention coefficients for neighboring data. Hao et al. (2019) showed that attention layer is also able to deal with longer sequence of input. They developed a sequence-to-sequence learning model that incorporated an attention layer and utilized a stacked BI-LSTM network as an encoder and another BI-LSTM network as a decoder. The LSTM helped to take input of longer sequences and the attention layer helped to remember only the important part of the sequences.

Additionally, recent study by Xinglei Wang et al. (2020) showed that by capturing longer temporal dependencies, the use of a convolutional network as an encoder can improve model performance. They used a seq2seq graph convolution architecture, which combined an attention layer with a convolutional encoder and a recurrent decoder. K. Zhang et al. (2020) proposed a Multi-Agent Attention Model to solve multi-vehicle routing problems. After rigorous offline training, their model solves routing problems instantly using an encoder-decoder framework with attention layers.

To investigate conflicts involving diagonal-crossing motorcycles, Yao et al. (2021) developed an interaction-aware multiple layered deep-learning framework. The framework includes an LSTM network as an encoder for extracting historical motion features. The intentions of drivers toward target lanes were captured by a vehicle intention summarizer was used. A Graph Attention Network (GAT) was used for modeling the interactions of left turning

---

<sup>3</sup> tf-seq2seq is a encoder-decoder framework for Tensorflow that is used for Machine Translation, Text Summarization, Conversational Modeling, Image Captioning, and more. <https://google.github.io/seq2seq/>

vehicles. A Vehicle-Motorcycle Interaction Extractor was applied to characterize the interaction from diagonal-crossing motorcycles on left-turning vehicles. The framework also uses a Vehicle Motion decoder that combines the outcomes of the preceding layers to generate future motion for left-turning vehicles.

C. Li et al. (2021) used attention mechanism with LSTM to capture temporal information and useful knowledge shared by other modes for demand prediction for multiple public transit modes. J. Sun & Kim (2021) used LSTM with self-attention to predict future locations and travel times simultaneously. The self-attention layer learns correlations between distant positions in a single sequence of data points. Xuesong Wang et al. (2022) identified distracted driving using Bi-LSTM with attention layers, in which the attention layers were added between the LSTM and final dense layers. The attention layers were used to capture driving behavior patterns that were highly correlated with phone use in the trajectory as well as vehicle dynamics of naturalistic driving data.

#### 2.2.3.2 *Attention Layer in Text Analytics*

Attention-based model is now a widely used techniques in natural language processing for product review and sentiment analysis<sup>4</sup>. For example, Zhai et al. (2020) used local and

---

<sup>4</sup> Sentiment analysis is the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications.

global attention layer with BI-LSTM encoder for sentiment analysis. Their global attention layer calculates attention weights for all the words in the sentence and the local attention layer looks only subset of words in the sentence. The subset of words was selected using syntactic-based distance from the target words. Dong et al. (2020) proposed a text classification method that combines attention layer with joint embedding learning of labels and words to capture the interaction between all the sentences in a text. The model provides text representations with comprehensive semantics. K. Sun et al. (2019) used an attention-based CNN model to extract sentence features and classify relationships in texts. They used CNN to extract sentence-level features and inner-attention to extract sentence-level features for the model. The BI-LSTM was used as an encoder to capture semantic information within sentences. It was fed into the attention layer to produce better dependency relationships in the sentences, and connected to final output via CNN layers. Du et al. (2019) developed an advance DNN, which is a convolution-based attention with BI-LSTM sentence encoder for sentiment analysis. Their model outperformed Hierarchical Attention Network (HAN) of Z. Yang et al. (2016) in documents with more than 20 sentences. H. T. Zheng et al. (2017) developed attention based model to generate news comment. Zhu et al. (2019) used GRU with attention layer for relation classification between two entities in a sentence, in which a merge attention mechanism was applied to capture the correlations among relations. Feng et al. (2022) applied masked attention layer that discards noisy words from the sentences based on a threshold value. Liu et al. (2020) applied syntactical attention and the

---

Affective states: affective states are longer lasting mood states (such as anxiety or depression) which are not caused by a single stimulus but are the results of an accumulation of experiences. In psychology, emotions (affective states) are complex psychophysiological constructs composed of many underlying dimensions. Highly positive affective states include enthusiasm, confidence, happiness, alertness, etc. Highly negative affective states include anger, disgust, fear, guilt, etc..

semantical attention mechanism with BI-LSTM and fed them into a ConvNet for text classification. Instead of calculating weighted sum, which is the more traditional method to compute attention score, they generate matrix to retain order information. P. Wang et al. (2021) developed a sentence-to-sentence attention model using multiheaded attention layer for sentiment analysis of online review. Apart from BI-GRU, their model outperforms and converges faster than the standard model such as CNN, LSTM, and RNN. Significant differences in model accuracy are found among different sources and target domains.

In summary, the attention layer-based model has been widely used to solve a variety of real-world problems, including transportation problems. The BI-LSTM is the most widely used RNN, having been tested with a wide variety of attention layers and showing promising results. A BI-LSTM reads a document from both directions (start to end and end to start), which helps in the extraction of more accurate contextual meaning than a traditional LSTM, which reads the document only from start to end. Another key observation is that attention layer can capture both the patterns within the same sequences (temporal relationships), and the patterns between sequences that are related geographically (spatial relationships). The takeaways from this review assisted in presenting the extent and scope of deep neural network (DNN) and the applicability of attention layer in transportation text analytics.

### **2.3 Critical Findings in Existing Studies**

Various techniques, ranging from manual review to NLP-based deep neural networks, are used to formulate policy and make decisions, as well as to address transportation-related issues.

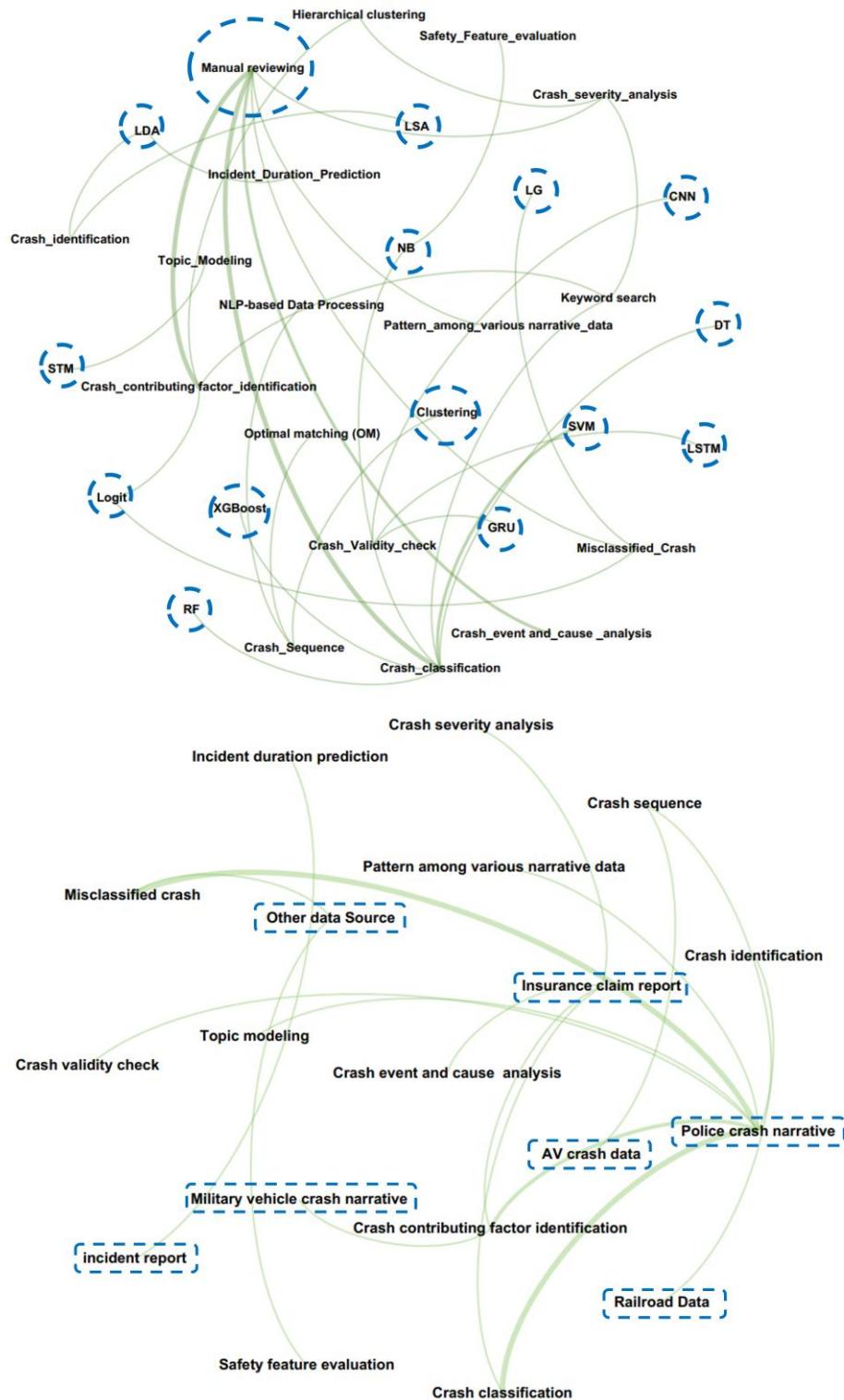
Table 2.3-1 presents the research objectives, various text analytic methods and data from existing research. It showed that most studies are focused on crash classification, crash contributing factor identification, crash severity and cause analysis, and crash event analysis. Several studies applied conventional text mining techniques, while a few used shallow machine learning techniques. The most frequently used conventional text mining text mining techniques are Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). The shallow machine learning includes hierarchical clustering, k-means clustering, probabilistic clustering, XGBoost, naive bayes, nearest neighbor, and decision tree. In the transportation field, the use of deep learning as text analytics is insignificant.

**Table 2.3-1 Summary of Literature Review**

Research Objectives	Methods	Data
Crash classification	Manual Reviewing	Police crash narrative
Crash type classification	Keyword search	Insurance claim report
Crash contributing factor identification	Latent semantic analysis (LSA) Latent dirichlet allocation (LDA)	Social media data Medical data
Crash severity analysis	Hierarchical clustering	Autonomous vehicle crash data
Crash event and cause analysis	K-means clustering	Survey data
Crash pattern analysis	Naive Bayes Logistic regression Decision tree Support vector Machine XGBoost Recurrent Neural network Convolutional Neural Network	

The network diagram (Figure 2.3-1) was used to determine which research problems, methods, and sources of text data in transportation domain have received the most attention. The node represents each distinct element of research, such as research objectives, methods, and data, and the line represents their relationship. The width of the line represents the frequency of research. The diagram presents that many researchers used manual methods (e.g., reviewing crash reports or preparing list of keywords by experienced person) to determine crash factors and injury severity. However, manual methods are quite inefficient and time consuming, when working with large datasets.

Other studies focused on information extraction techniques. It is more common to apply shallow machine learning techniques for classification problems. Shallow machine learning techniques generate outcomes by using words and word-frequencies, also known as bag of words (BOW). BOW ignores word contexts because it does not consider word order. Therefore, the BOW method is ineffective, especially when the role of words varies across contexts.



**Figure 2.3-1 Research Objectives vs Methods (Top) and Research Objectives Vs Data (Bottom)**

Interestingly, the use of DNN for textual data analysis in the transportation field is very insignificant compared to other domains. DNN, particularly RNN (e.g., LSTM, GRU) has been widely used techniques as text analytics. Although RNN captures the contexts of words, it uses the entire report as an input vector, making it ineffective for classifying longer reports. In other words, when encounter with lengthy reports, GRU or LSTM alone is incapable of memorizing context. Attention layer combined with DNN has recently garnered considerable interest in other research domains. It is a widely used technique due to its interpretability of the results. It can detect and learn the important parts of the text through utilizing the context of the words and sentences, which differentiates it from traditional DNN models. In the transportation field, attention-based models have been used for forecasting, prediction, and object detection using numeric or video data. Surprisingly, no research has utilized attention layers for text data.

Based on the aforementioned discussion, this dissertation identified the following research findings in existing research:

1. *Classification.* A significant amount of research has been conducted for text classification. The two most popular techniques are expert-prepared keyword lists and crash report reviews. In this case, automatic keyword extraction is required to replace the need for expert knowledge.
2. *Crash report.* Existing research revealed that crash narratives contained valuable information that could be used to identify important crash factors or classify a new crash type (such as secondary crashes). However, none of the research investigated the utility of crash narratives to improve structure data.

3. *Machine learning.* In several transportation research, shallow machine learning has been applied to crash narratives. However, none of the research has conducted a cross-analysis of the model outcomes to determine their relative utility.
4. *Attention layer.* Besides attention layer-based text analytics research in other domains, the utility of attention layers using structure data has been recognized in the transportation field. However, attention layers have never been used for analyzing transportation text data. Despite shallow ML techniques have several limitations to analyze text data, the research in transportation is mostly limited to shallow ML techniques.
5. *Facilitate narrative investigation process.* Text analytics involves complex data processing and visualization techniques. To facilitate the benefits of text analytics research, it is common to make the research readily available in a form of tool or software. Even though several studies showed intuitively the need for software to review and investigate the crash narratives, none of the study showed inclination to develop such a software.

Considering the above findings, this dissertation will investigate crash narratives in detail, develop algorithms to automatically extract information from narratives, and discuss the results in the context of improving traffic safety. Additionally, this study will develop a crash investigation software to facilitate crash investigation using text data.

## **Chapter 3. RECOVERING MISSED CRASHES FROM CRASH**

### **NARRATIVES- A NOISYOR METHOD**

#### **3.1 Introduction**

Motor vehicle accident reports are the most useful and valuable source for analyzing crashes and identifying crash contributing factors. Most information of a crash is filled in by law enforcement agencies in appropriate data fields and later stored in a database, so called structured data. A significant amount of information is also presented in unstructured text such as crash narratives. A crash narrative is the detailed description of a crash by law enforcement officers. The narrative fields can be used to record additional information on specific circumstances and key factors (e.g., citations, additional witnesses, types of drugs and medication, hazardous materials spillage from trucks and buses, trailer and towed, school bus information) and other possible crash contributing circumstances not available as structured data fields. More importantly, the narrative provides detailed explanations to these contributing circumstances such as driver, vehicle, or highway, and often times, specific crash location information. Unstructured data can't be easily stored in a database. And even if it is stored, it has attributes that make it difficult to edit, query and analyze, especially on the fly.

However, mistakes can happen in a crash report such as misclassification. Failing to classify a crash to the appropriate category will lead to undercounting some types while overcounting others. Crashes are missed in the structured data field for a variety of reasons: restrictive reporting options in tabular forms (Blackman et al., 2020; Ullman & Scriba, 2004; J.

Wang et al., 1996); lack of understanding about the importance of the crashes, overloaded by work during crash reporting time (Graham & Migletz, 1983); and misclassification of crashes (Wang et al., 1996, Farmer, 2003). Generally, a police officer makes certain judgments about a crash based on the severity of the crash and the driver. A fatal crash is usually given the highest reporting priority, compared with property damage crashes which usually receive a lower priority (Ye & Lord, 2011b). A crashes with less severity or no injuries are not reported in structured data (J. Wang et al., 1996). In addition, the probability of reporting an injury crash increases with the age of the injured person(i.e., for young children, it is 20-30%; and for persons over 60, it is 70%); and the number of vehicles involved. (Ezra Hauer & Hakkert, 1988). A crash involving a younger or female driver has a lower probability of being reported (Amoros et al., 2006). Estimates based solely on structured data fields underestimate the results of the safety analysis (Abay, 2015).

Law enforcement officers' narratives can use different words or phrases, which presents a challenge for traffic safety engineers when querying specific terms. And while engineers often manually review the reports to search for causes and contributing factors for remedial actions, the process is labor intensive, and the review quality is inconsistent, as it is subject to the reviewers' experience and judgement. Automatic information extraction through text mining techniques is predictable, consistent, and efficient. A crash narrative can be converted to a numeric vector suitable for machine learning, a process often referred to as feature extraction. Text mining results can be used to assess the quality of crash flags (e.g., work zone, secondary crash). Moreover, reviewing thousands of crash reports is a matter of minutes, according to a recent study. The purpose of this research is to develop and implement a NoisyOR-based text

mining algorithm for resolving a critical transportation safety problem known as recovering missed crashes. Two case studies have been used to illustrate the issues. One objective is to identify missed work zone crashes based on the crash narrative, while the other objective is to identify missed distracted/inattentive driving-related crashes based on the crash narrative. In the second case study, distracted driving crashes are distinguished from cases of inattentive driving. The results of all models are summarized and compared; their performance is discussed; and the models that are most appropriate are recommended.

### **3.2 NoisyOR-based Classification**

In the NoisyOR method, the probability of being a specific type of crash narrative is calculated by combining the probability scores of unigrams (words) and bigrams (two consecutive words) in the narrative. It is a probabilistic extension of logical “or” (Onisko et al., 2001; Vomlel, 2006). If any input has a high probability score (such as a value close to 1) then the combined probability in NoisyOR becomes high. The combined probability in NoisyOR is even higher if more input probabilities are high. To apply NoisyOR classifier to crash narratives, it is needed to compute the probabilities of unigram, bigram, and trigram and combining these probabilities using the NoisyOR method, which are discussed in the following section.

### 3.2.1 Equation of Simple Count Probability

For every unigram, bigram, and trigram  $w$  in the corpus, the method first computes the probability that if it is present in a narrative, then the narrative is positive, i.e.,  $P(\text{positive})$ . Then, this probability is computed using simple frequency counts, as shown in Equation 1.

$$\text{Probability Score } (w) = \frac{\text{Positive Count}(w)+1}{\text{Positive Count }(w)+\text{Negative Count}(w)+2} \quad (1)$$

where  $w$  is a unigram, a bigram, or a trigram. *Positive Count* means the number of occurrences of  $w$  in the positive narratives. Similarly, the *Negative Count* indicates the number of events of  $w$  in the negative narratives.

The equation essentially computes out of all the narratives in which  $w$  occurs how many narratives are positives, which is the probability that a narrative will be positive if  $w$  occurs in it. Then, smoothing is applied by adding one in the numerator and two in the denominator of the equation. This simple version of Laplace smoothing assumes  $w$  occurred at least once in a positive narrative and a negative narrative. Smoothing done in this way ensures that among the unigrams, bigrams, and trigrams that have zero negative counts, the ones with higher positive counts receive higher probability scores. Otherwise, they will all receive an unrealistic probability score of 1 because they occurred in a few positive narratives and no negative narratives.

In case of the words that appear in both positive and negative narratives with very high frequency (Count), it is likely to reduce the probability of that specific word. For example, if a unigram ‘*unit*’ appears in the narratives of a specific type of crash (positive case) 110,933 times

and in all the other narratives (negative cases) that excludes that specific crash 1,000,904 times, which according to Equation 1, gives a probability of 0.099. It indicates that the word is not relevant for the classification task. On the other hand, if a unigram/ bigram/ trigram appears in both positive and negative narratives with high frequency but has a higher frequency in positive narratives, Equation 1 gives a good probability score to the corresponding unigram/ bigram/ trigram. For example, if a unigram '*inattentive*' appears in the narratives of a specific crash 2743 times and in all the other narratives 1808 times, which according to Equation 1, gives a probability of 0.6023, indicating that the word is relevant for the classification task.

To classify a given narrative as positive or negative, its probability of being positive is computed by combining the probability scores of the unigrams, bigrams, and trigrams present in it. The method needs to compute  $P(\text{positive} | w_1, w_2, \dots, w_n)$ , where  $w_1..w_n$  are unigrams, bigrams, and trigrams present in the narrative. It computes it by combining the probabilities  $P(\text{positive}|w_1)$ ,  $P(\text{positive}|w_2), \dots, P(\text{positive}|w_n)$ , which have been computed as described earlier. NoisyOR is a method of combining probabilities (Zagorecki & Druzdzel, 2004), which is commonly used in Bayesian networks (Onisko et al., 2001; Vomlel, 2006). Instead of true/false values in Noisy-OR, the inputs and output are probabilities (hence termed "noisy"). Analogous to logical "or", in Noisy-OR, if any one of the input probabilities is high (i.e., close to 1), then the combined probability is high. But unlike logical "or", the combined probability is even higher if more input probabilities are high. The combined probability is low (i.e., close to 0) only when all the input probabilities are low. NoisyOR combined probability is mathematically computed as shown in Equation 2, where the probability score of a narrative is calculated by combining the probability scores of unigrams, bigrams, or trigrams occurring in it.

$$\text{NoisyOR Probability Score } (N) = 1 - \prod_{i,j=1}^n (1 - P_i)^j \quad (2)$$

where N is a given narrative,  $P_i$  indicates the probability score of  $i^{\text{th}}$  unigram, bigram, or trigram as computed from the training data using Equation 1, and j means the number of occurrences of that  $i^{\text{th}}$  unigram, or bigram or trigram in the crash narrative N. It should be clear from Equation 2 that if there is no unigram, or bigram, or trigram in a narrative with a high probability score, then the probability score of the narrative will be close to zero. On the other hand, a single unigram, or bigram, or trigram with a high probability score will result in a high probability score of the entire narrative. This fact is precisely the behavior that has been observed in the data. Furthermore, more unigrams, bigrams, and trigrams with high probability scores make the combined probability score higher.

### **3.2.2 Equation of Weighted Count Probability**

The probability scores computed using Equation 1 will be adversely affected if the number of negative narratives is disproportionately higher than the number of positive narratives. In the Weighted Count Probability equation, the positive counts are weighted by the average number of positive word appearances in the positive narratives. It is designed to capture the unigrams/bigrams/trigrams that appear not only more often in positive narratives than negative narratives, but also more often per positive narrative than per negative narrative. The Weighted Count Probability is formulated in Equation 3:

*Probability Score (w) =*

$$\frac{\text{Positive Count}(w) * \left( \frac{\text{Positive Count}(w)}{\text{Number of instances in positive}} \right) + 1}{\left( \text{Positive Count}(w) * \left( \frac{\text{Positive Count}(w)}{\text{Number of instances in positive}} \right) \right) + \left( \text{Negative Count}(w) * \left( \frac{\text{Negative Count}(w)}{\text{Number of instances in negative}} \right) \right) + 2} \quad (3)$$

Here, *Positive Count* and *Negative Count* represent the same meaning as in Equation 1.

*Number of instances in positive* means the total number of reported cases for distracted or inattentive. The *Number of instances in negative* means the total number of cases not reported as distracted or inattentive.

With the probability scores obtained using Equation 3, the probability of a positive narrative is computed using the NoisyOR method described earlier. Given that a positive narrative will have an indicative word mentioned more than once, the NoisyOR probability score of the narrative will increase accordingly (note that in Equation 2,  $(1 - P_i)$  is raised to the power of  $j$ , the number of occurrences). In contrast, a negative narrative that has fewer indicative words will have a lower probability of being positive.

### 3.3 Case Study 1: Work Zone Crashes

Work zone activities are essential for maintaining good roadways, supporting economic development and competition, and improving safety. While road work is temporary, the poor decisions and mistakes made by motorists that lead to work zone crashes can have lasting impacts. According to the Federal Highway Administration (FHWA), 27,037 people, or 773 per year, died in work zone crashes in the U.S. from 1982 through 2017 (CDC, 2020). In Wisconsin, more than 2,600 work zone crashes took place every year over the past five years, resulting in

5,200 injuries and 50 deaths (WisDOT, 2020). Work zone safety for both motorists and workers is an urgent issue that must be addressed through better design, operations and management.

Work zones near traffic, whether they involve major road construction, utility work, or emergency vehicles at the side of the road, always present some risk to both drivers and workers. Identifying and analyzing historical work zone crashes can save lives.

Observational safety analysis has been instrumental in identifying potential deficiencies in work zone design and traffic operations. Examples of safety analyses based on crash data include: crash rate estimation across different work zone configurations (Cheng et al., 2012; Daniel et al., 2000; Elias & Herbsman, 2000; Khattak et al., 2002); crash pattern identification and categorization(Garber & Zhao, 2002; Graham et al., 1978; Weng et al., 2016); work zone crash prediction (Y. Li & Bai, 2009b; Meng et al., 2010); and evaluating the safety of innovative work zone designs and management strategies (Y. Li & Bai, 2009a; Maze et al., 2005; Rahman et al., 2017; Ullman et al., 2008). All the aforementioned examples are dependent on the completeness and accuracy of work zone crash data. The crash in the structured data may not have been coded or recorded as that specific crash type.

### **3.3.1 Data Collection**

The dataset comprised 377,479 crash reports that occurred between January 1, 2017 and October 31, 2019 that were acquired from the Wisconsin Department of Transportation (WisDOT) through the WisTransPortal data hub. A construction zone flag (CONSZONE) within the crash data indicates whether “*a crash occurred in a construction, maintenance, or utility*

*work zone or is related to activity within a work zone*”. The reported work-zone (WZ) crashes make up 2.27%, 2.49%, and 1.93% of total crashes for years 2017, 2018 and 2019, respectively. Narratives were included in 94.21% of the reported WZ crashes and 77% of the non-work zone (NWZ) crashes. The ratio of WZ to NWZ crashes is 1:36, which is a highly imbalanced dataset.

The two following sample crash narratives were randomly chosen from the dataset to illustrate the structure of crash narratives.

WZ crash narrative example:

“Entering construction zone with right lane closure. Unit 1 driver stated unit 2 and a semi were straddling center line. Unit 1 driver stated thought unit two was merging to right lane toward hwy c exit and tried to pass unit 2. Unit 1 driver stated himself and semi were straddling traffic lane to stop other drivers from passing on right as right lane was closed ahead. Unit 2 stated unit 1 attempted to pass on left shoulder but ran out of room due to portable warning sign. Unit 2 driver stated unit 1 driver side swiped driver side.”

NWZ crash narrative example:

“Unit #2 was stopped in the inside straight lane of eastbound university ave., at a red light at the intersection with n. Midvale blvd. Unit #1 was traveling in the same lane directly behind unit #2, and was unable to stop in time to avoid a rear end collision with unit #2. The roadway was wet, and the weather conditions were rainy.”

The numeric values within the narratives usually represent date, time, driver, and road information. The narratives have a certain formality but can still be flexible in the sequence of events. In the WZ narrative, some sentences contain words that indicate WZ (e.g., “construction zone”, “right lane closure”, “portable warning sign”), while others do not contain any WZ indicators. In fact, the latter cannot be distinguished from sentences that could have been in a NWZ narrative. This observation is true of other WZ narratives as well; only a few words are indicative of a WZ while the rest of the narrative is not, suggesting that presence of just a few

words can be used to identify a WZ narrative without having a deep understanding of the entire narrative. Additionally, there are no such words in the narrative that specifically indicate NWZ.

### *3.3.1.1 The Nature of Noisy Data*

In this study, the 2017 and 2018 work zone crash data were used to train a classifier (described later) and the NWZ narratives of 2019 (Data was available till October 31, 2019) were used as test data to recover missed WZ crashes. The narratives corresponding to reported WZ crashes (i.e., marked under CONSZONE flag) were used as examples of WZ narratives to train the classifier. Similarly, the narratives corresponding to reported NWZ crashes (i.e., not marked under CONSZONE flag) were used as examples of NWZ narratives. The method did not require the manual annotation of training examples, a task that usually requires the huge effort of training a classifier.

However, the training dataset created does include a high level of noise. On one hand, many narratives of reported WZ crashes may not have any relevant information about the WZ. For example, the officer may have already indicated a crash as WZ by using the CONSZONE flag, hence not feeling the need to mention it in the narrative. However, WZ crashes are known to be missed, and there are narratives corresponding to reported NWZ crashes that are actually WZ. The classifier may have difficulty learning from such noisy training data.

### *3.3.1.2 Data Cleaning and Pre-processing*

Several text mining techniques for data cleaning and pre-processing were applied to prepare the data. The key terminologies from the text mining domain are introduced here:

- Corpus is the collection of all of the narratives.
- Tokenization is the process of breaking up the sentence into a token. A token can be words, numbers, unigram, or bigram. The terms unigram and bigram are used interchangeably as the token in this study.
- Collection frequency (cf) is the number of times a token occurred in the corpus.
- Term frequency (tf) is the number of times a token occurred in a narrative.
- Document frequency (df) is the number of documents/narratives that contain a token. Only the tokens with high df values in WZ narratives will have a high impact on the model.

In the training dataset, the narratives were first lower-cased to merge the occurrences of the same word in different cases. Then, all punctuations and special characters (e.g., ! " # \$ % & ' ) \* +, -./:; <=>? @ [ ] ^ \_ ` { | } ~) were removed from the narratives. Next, the narratives were converted into tokens to build a vocabulary list from the training set. The narratives may include spelling errors and/or words in multiple forms, such as “zone” and “zones” or “construction” and “construct”, which are common issues when mining unstructured text data. While some text mining techniques can handle these issues, there is no guarantee the problem

will be solved completely. Furthermore, improper processing of these words may lead to new problems. Thus, the words in the vocabulary list were kept as-is.

### **3.3.2 Crash Verification and Model Performance Evaluation**

The main objective of this case study is to find missed WZ crashes from the crash narratives. The performance of the proposed method on the test dataset was manually reviewed. Since the test data are unlabeled, it is not possible to manually check all possible WZ crashes from the huge test data (over 80,000 cases). Initially, it was anticipated that a threshold could be set up for the classification score to separate the cases into WZ and NWZ. However, the classification scores show that many cases have very small differences. Hence, the performance of the model was evaluated by sorting the results in descending order so that the most probable scenarios are at the top. A manual review was conducted to manually classify (NWZ or WZ) the top 100 narratives with the highest probability scores using only unigrams, as well as the top 450 narratives with the highest probability scores using both unigrams and bigrams. Each reviewer was assigned an equal number of samples to eliminate any reviewer bias. This study avoids using any external data (such as work zone inventory data) for evaluating the model performance.

### **3.3.3 Results and Analysis**

The results of NoisyOR with unigram and unigram +bigram is compared and discussed in this section. The characteristics of missed crashes are analyzed from spatial and temporal

perspectives, along with other features. The additional analysis is expected to provide insight on the circumstances in which crashes are not reported as WZ related so that recommendations can be made for improving future data collection.

### *3.3.3.1 The Analysis of Positive Unigram and Positive Bigram*

The 2017-2018 crash data were cleaned and preprocessed, showing 10,875 unigram and 96,550 bigram words (tokens) in the corpus. Table 3.3-1 presents the top ten positive unigrams and bigrams and their corresponding probability scores. As shown in Table 3.3-1, the bigram approach extracts more WZ-related information than the unigram approach. However, despite high probability scores, some positive unigrams did not carry meaningful information such as “Kampo”, “Kucej”, or “Werych”. While “Kampo”, “Kucej”, and “Werych” may appear only in WZ cases, at a very low frequency, meaning including them in the Positive Unigram list may degrade the model’s performance. For example, if a narrative has many such unigrams, the NoisyOR may tend to classify it as a WZ crash even if it’s not.

The document frequency (df) and collection frequency (cf) of the training set were calculated to examine how the positive unigrams and bigrams with high probability scores influence the proposed method. The classifier performance did not degrade much due to lower document frequency(df) of the less meaningful positive unigrams and bigrams. Thus, an important positive token should have both high df and cf values and with high probability score.

**Table 3.3-1 Top Ten Positive Unigrams and Bigrams by Probability Score Using Equation 1**

Rank	Positive Unigram		Positive Bigram	
	Positive Words	Probability	Positive Words	Probability
1	flagman	0.960	active construction	0.990
2	taper	0.947	in construction	0.988
3	barreled	0.937	temporary cement	0.983
4	dividers	0.929	zone where	0.972
5	roadworks	0.923	construction crew	0.971
6	kampo	0.917	zone lane	0.964
7	unfinished	0.917	interstate is	0.960
8	flaggers	0.917	no workers	0.960
9	kucej	0.909	flag person	0.957
10	werych	0.900	workers present	0.956

Table 3.3-2 populates a list of the top 15 important positive unigrams and bigrams ranked by df, cf and probability score ( $p_r$ ) in a decreasing order. In the positive unigram list, the token “construction” is the most important because it has the highest df and cf values. The most important token in the positive bigram list is “construction zone”. Approximately 35.15 % of the WZ crash narratives contain the token “construction”, whereas 16.16% of WZ crash narratives contain “construction zone”. Table 3.3-2 shows that the positive bigram list offers more specific WZ crash information and higher probability scores than the unigram list.

**Table 3.3-2 Top 15 Positive Unigrams and Positive Bigrams By df, cf, and Probability Score**

Rank	Positive Unigram				Positive Bigram			
	Token	cf	df	Pr	Token	cf	df	Pr
1	construction	2960	2088	0.89	construction zone	966	826	0.9
2	zone	1181	972	0.45	the construction	763	625	0.82
3	closed	743	588	0.44	a construction	484	437	0.77
4	barrels	407	314	0.69	to construction	320	312	0.73
5	closure	265	191	0.61	was closed	242	228	0.51
6	orange	192	152	0.34	construction barrels	195	167	0.77

7	barrel	228	147	0.56	lane closed	212	161	0.67
8	temporary	170	126	0.37	construction unit	158	156	0.78
9	zoo	219	123	0.56	under construction	151	149	0.79
10	cones	166	122	0.45	construction area	158	136	0.82
11	workers	120	110	0.52	road construction	145	135	0.68
12	barriers	119	97	0.49	work zone	161	132	0.92
13	barricades	107	78	0.42	the zoo	206	120	0.67
14	attenuator	145	74	0.47	construction and	123	120	0.7
15	worker	95	67	0.51	zoo interchange	181	114	0.67

\* cf = collection frequency in WZ narratives, df = document frequency in WZ narratives,  $p_r$  = probability.

The unigram method will classify a narrative as a WZ crash if the narrative contains the token “construction” ( $p_r= 0.89$ ) from the positive unigram list only because the threshold value is greater than or equal to 0.89. The df of “construction” is much higher compared to other unigrams in the list, so the misclassification rate by the unigram method will be higher.

Compared with the positive unigram “construction”, the positive bigram “construction zone” ( $p_r= 0.90$ ) is more contextual and has a higher df than the remaining bigrams in the list. A narrative with the presence of “construction zone” instead of “construction” is more likely to be correctly classified as a WZ crash. The manual review result shows that all of the NWZ narratives that contain “construction zone” are true WZ crashes. However, 22 NWZ narratives that contain “construction” are not WZ crashes.

Positive tokens such as “fst”, “kampo” and “kicmol” in the positive unigram list do not carry any meaningful information. These unigrams have a small df with high probability scores, meaning they should be discarded to reduce the misclassification rate. The positive token lists

also contain names of locations such as “zoo” in unigram and “the zoo<sup>5</sup>” in bigram. The presence of those tokens can cause the NoisyOR method to misclassify NWZ crashes as WZ crashes.

### 3.3.3.2 Comparing Unigram and Unigram +Bigram Methods

The preceding section explains that in using the NoisyOR method, the unigram method may not be effective as expected. Positive unigrams with high cf values may have low probability values because the same unigrams also appear in the NWZ crash narratives. The problem can be mitigated by adding some context to the NoisyOR approach, such as in the form of bigrams. The ordered positive bigram list provides more contextual information related to WZ. This section provides empirical evidence of using the NoisyOR method as a text classifier to identify missed WZ crashes from narratives. The section also explores the classification outcomes of unigram and unigram +bigram when compared with gold label, or manual reviewing.

The 100 narratives with the highest probability scores in each classifier were manually reviewed. The top 100 narratives of the unigram NoisyOR classifier included 65 actual WZ crashes, while the top 100 narratives of the unigram +bigram NoisyOR classifier included 78 actual WZ crashes. The unigram +bigram NoisyOR narratives that were correctly classified

---

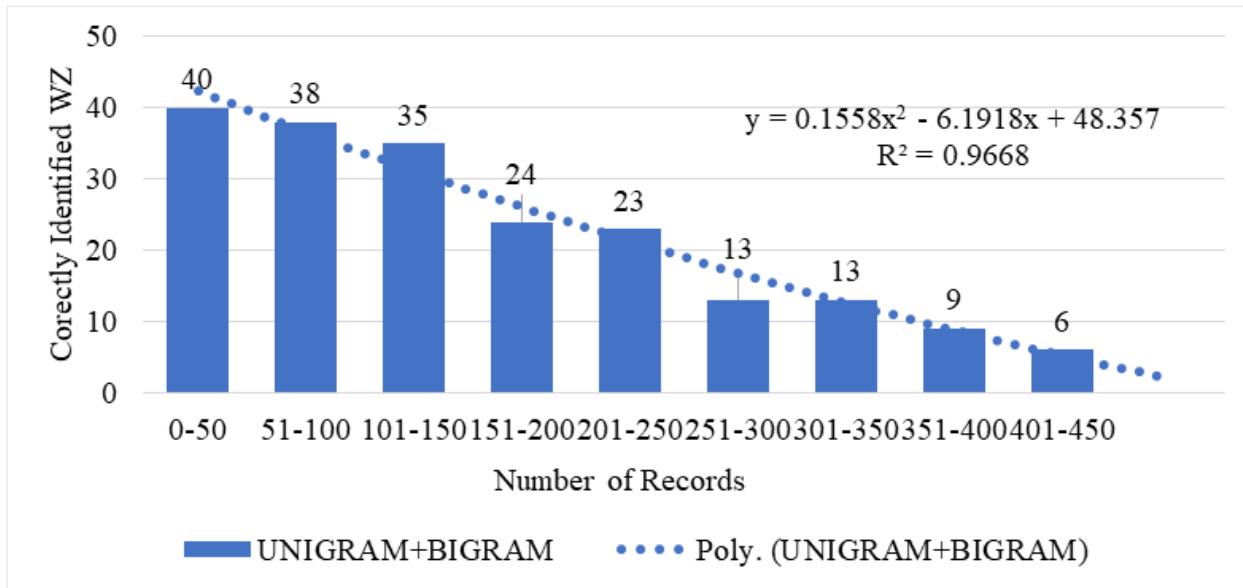
<sup>5</sup> Zoo interchange construction is the most complex and expensive highway project in Wisconsin’s history, which began in 2014 with an expected completion date of 2022.

contained more contextual positive bigrams such as “construction zone”, “under construction”, “construction worker” and “lane closed” with high df values in the WZ training set.

A close review of 35 unigram NoisyOR cases that were misclassified shows that they contain WZ-related positive unigrams such as “con-struction”, “barrels”, “attenuator”, “barricades”, “orange” and some noisy words such as “carrao”, “kampo”, “melloch”. These noisy unig-rams have high df values in the WZ training set, indicating their popularity in the WZ crash narratives. On the contrary, the unigram +bigram NoisyOR misclassified 22 cases from its top 100 narratives. A close re-view of these 22 cases reveals that the unigram portion of unigram +bigram NoisyOR contains few positive unigrams but with high probability scores; the bigram portion contains a longer list of positive bigrams with moderate probability values. Thus, the comparison reaffirms that unigram +bigram NoisyOR tackled the noisy tokens more successfully than unigram noisy-OR.

### *3.3.3.3 Extended Analysis of Unigram+Bigram of NoisyOR*

Further analysis was performed to quantify the classification accuracy rate against the case rank of the unigram+bigram method. Starting from the highest-ranked cases, the number of correctly identified WZ crashes is counted over the 50-case intervals, as shown in Figure 3.3-1.

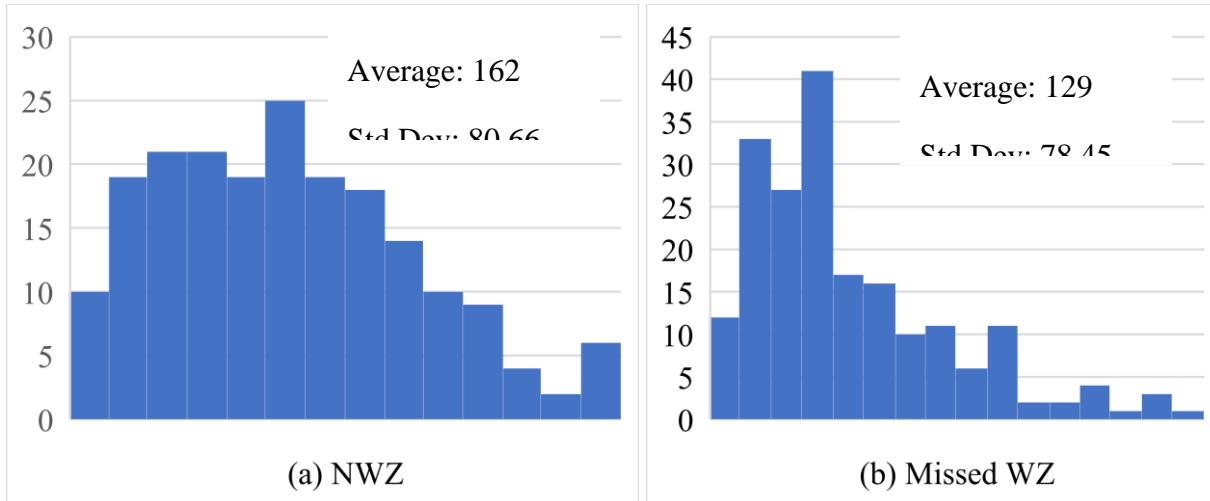


**Figure 3.3-1 Accuracy of (Unigram+Bigram) NoisyOR**

From Figure 3.3-1, two observations can be made based on the 450 cases reviewed: a) more than 50% of cases correctly classified till the fifth interval (201-250), and b) the model performance degrades rapidly from 80% in the first interval [0-50] to 12% in the last interval [401-450]. The fitted quadratic equation has a  $R^2$  value of 0.9668, suggesting a strong and consistent trend for the descending accuracy rate. The findings are good news for an agency who wants to estimate the effort of a manual review for missed WZ crashes.

The probabilistic distribution of narrative length was plotted for WZ and NWZ crashes, respectively, in Figure 3.3-2. The distribution was inspired by a study that shows that narratives not designated by officers as speed-related crashes have a longer length on average than non-speed related crashes (Fitzpatrick et al., 2017). Figure 3.3-2 shows that the narrative length of actual NWZ crashes is approximately normally distributed, while missed WZ crashes are slightly

skewed toward the left. The two distributions are statistically different at a 5% level of significance (two sample t-test,  $p=<0.0001$ ).

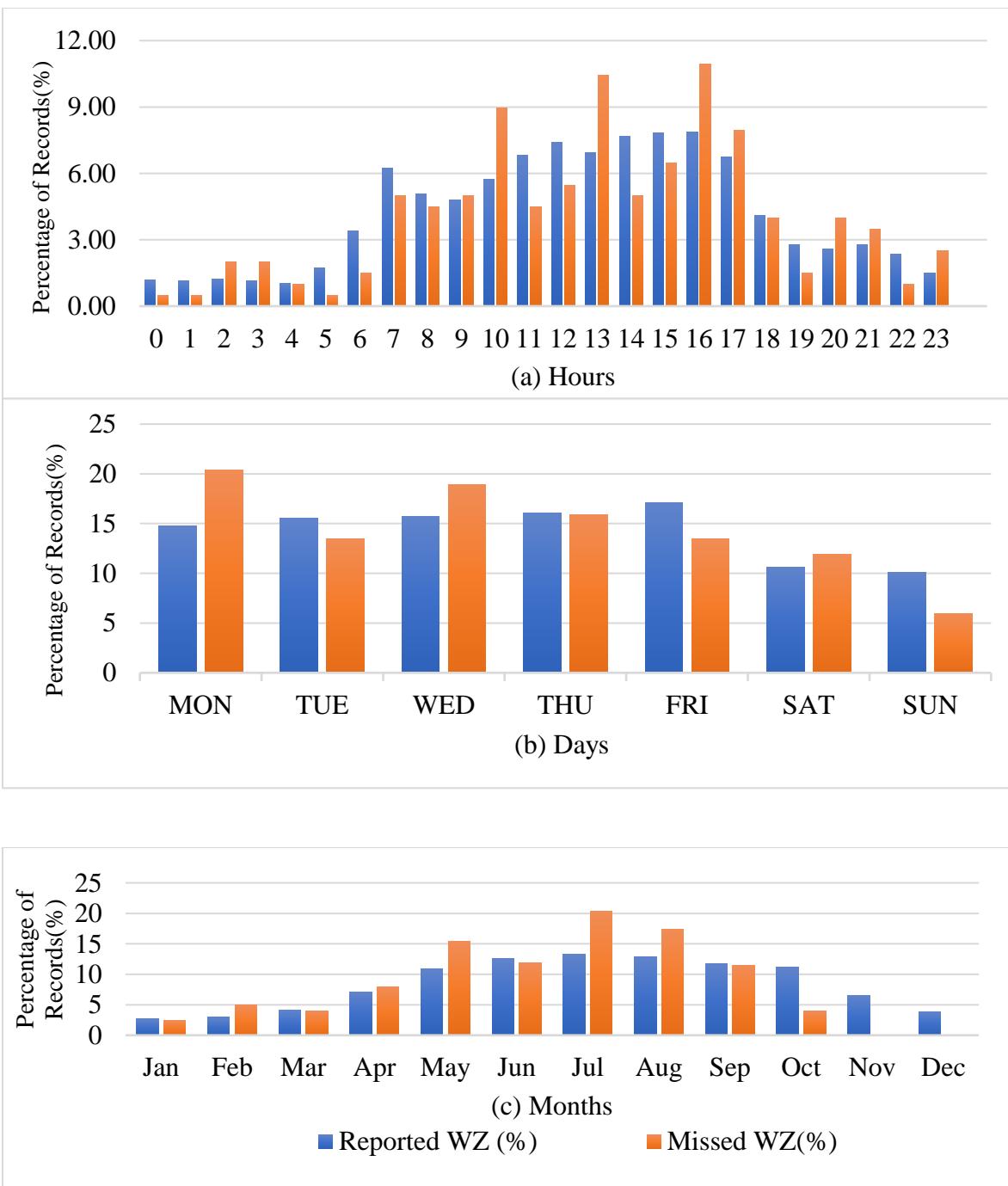


**Figure 3.3-2 Histogram of Narrative Length for a) NWZ and b) Missed WZ**

Moreover, the average narrative length of reported WZ crashes is 104, and Std. is 68 (sample size:1989), which is a statistically significant difference between NWZ (two sample t-test,  $p=<0.001$ ) and missed WZ (two sample t-test,  $p=<0.001$ ). Though it is expected that long narratives would have more positive tokens than short narratives, no correlations are observed between the length of narratives and the number of positive tokens for reported WZ and NWZ and missed WZ. In other words, there is not enough evidence to claim that long narratives tend to classify crashes more accurately than short narratives.

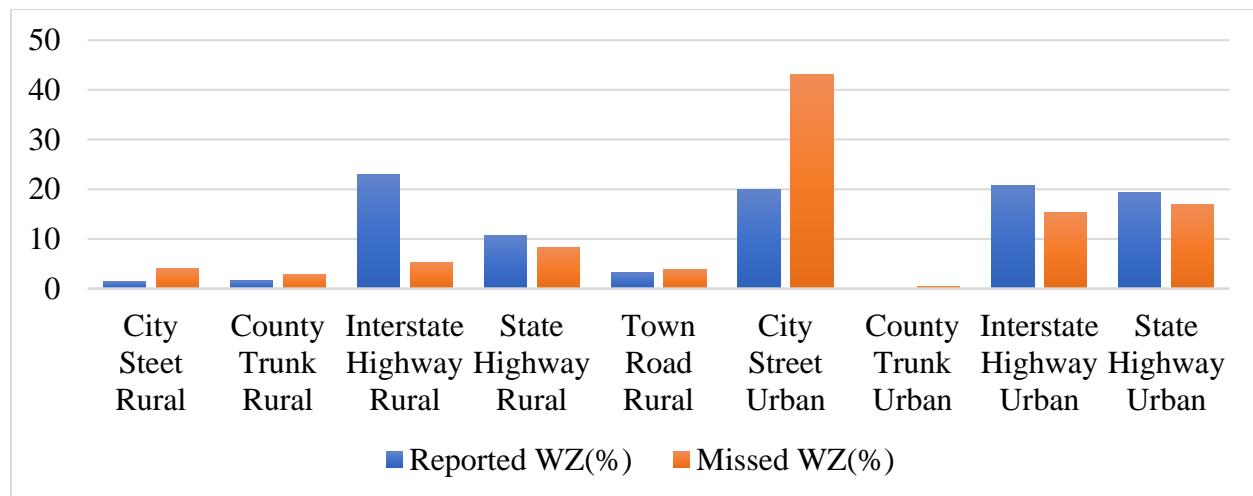
### *3.3.3.4 Analysis of missed WZ crashes*

Further analysis was conducted on the crash time and location for a better understanding of the circumstances under which a WZ crash is missed. Figure 3.3-3 shows the distribution of reported WZ and missed WZ confirmed in this study by time of day, day of week, and month of year. In 2017 to 2019, 70.96% of all reported WZ crashes and in 2019, 73.13% of the missed WZ crashes identified in this study occurred during daylight hours from 8 a.m. to 6 p.m., as shown in Figure 3.3-3 (a). Among daytime WZ crashes, a high percentage of missed cases occurred in the afternoon when traffic is busiest, from 4 p.m. to 5 p.m. It is plausible that crashes are missed when traffic is high or when construction activities are intense. The day of week distribution suggests that the WZ crashes are probably missed throughout the week, especially on Monday and Saturday, as shown in Figure 3.3-3 (b). Figure 3.3-3 (c) also displays the monthly distribution of reported WZ crashes versus missed WZ crashes, showing that a high percentage of missed cases are observed in the summertime, especially in July and August when construction activities are extensive and intensive.



**Figure 3.3-3 WZ Crash Analysis by a) Hour b) Day and c) Month**

Figure 3.3-4 shows the distribution of missed WZ crashes compared to reported WZ crashes by highway class. The evidence shows that most missed WZ crashes occurred in urban areas, including urban city streets (43.11%), urban state highways (16.89%) and urban interstate highways (15.33%). The interstate highway system, both urban and rural, has the best performance in terms of the low ratio of missed crashes to reported crashes. The next best performance is from state highways, where the ratio is close to 1. City streets have the highest ratio of missed crashes to reported crashes, particularly urban city streets which have only 20% of the total reported WZ crashes but make up 43% of missed WZ crashes identified in this study. Cheng et al. stated that construction work zones are usually assumed to be long term works, but maintenance or utility works are usually short term and temporarily, which may not be known to driver in advance (Cheng et al., 2012). Since many crashes on urban streets involve utility work zones, it is plausible that police may not consider those as construction zone related.



**Figure 3.3-4 WZ Crash Analysis by Highway Class**

Comparisons were conducted for other structured data fields, including weather conditions, pavement conditions, light conditions, and injury severity. The results show similar distributions between all reported WZ crashes and missed WZ crashes, mainly due to the lack of variety since most WZ crashes, reported or missed, occur during clear or cloudy weather, on dry pavement, in the daytime, and involve less severe injuries.

An analysis of missed cases suggests that 73.13% of the missed WZ crashes identified in the study occurred from 8 a.m. to 6 p.m. with a high percentage in the afternoon from 4 p.m. to 5 p.m. A high percentage of WZ crashes that are misclassified are observed in July and August when the construction activities are extensive and intensive. 43% of the missed WZ crashes identified in this study occurred on urban city streets.

### **3.3.4 Summary of Observations**

For this case study, a keyword-based text classifier was developed using the NoisyOR combined probability to identify misclassified WZ crashes from the crash narratives of police reports. Specifically, the unigram +bigram NoisyOR classifier was created and proven to be an effective means to recover WZ crashes from those police officers did not flag as Accident Analysis and Prevention 159 (2021) 1062117construction zone crashes. The narrative in a flagged WZ crash may not contain any relevant information linking it to a work zone, and narratives from unflagged WZ crashes may contain information related to a work zone. The NoisyOR method was used because of its ability to work effectively despite the high level of

noise in the unstructured text or crash narratives. Moreover, NoisyOR does not require much training time, is computationally efficient and is easier to implement.

A manual review of the top 450 cases classified as WZ crashes in the testing data recovered 201 missed WZ crashes, which was 0.24 % of the testing data. The review also indicated that beyond 450 cases, the chance of having missed WZ crashes may be very low. A follow-up analysis revealed that 73.13 % of the missed crashes occurred from 8 a.m. to 6 p.m., with a high percentage happening from 4 p.m. to 5 p.m. A large percentage of those crashes occurred in the summer (July and August) and 43 % occurred on urban city streets. The narratives of the cases that have high NoisyOR scores but are not WZ crashes were carefully reviewed and categorized into the five following groups:

- 1) Cases with positive words for location or address such as “the Zoo”, “Zoo interchange”: This issue is caused primarily by major roadway construction projects that span over multiple years, multiple stages and phases and multiple areas.
- 2) Cases with positive words for (temporal) traffic control devices such as “concrete barrier”, “median cement”, “attenuator” and “barriers”: Many of these devices, such as median concrete barriers, are permanently deployed to channelize traffic or to protect overpass and underpass structures such as an attenuator near a bridge or at a gore area.
- 3) Cases with weak positive words for traffic situations such as “congestion” or “backup” which are caused by non-WZ events (i.e., regular congestion or secondary crashes).

4) Cases with strong positive words such as “orange construction” or even “construction zone” whose situations are actually not related to a work zone location or work zone activities.

5) Undecided cases, even after a manual review: The authors were conservative and categorized undecided crashes from this study as NWZ crashes.

A location and/or time that a work zone crash occurred can certainly improve WZ classification in types 1 and 5. Such information, however, has to be linked to and retrieved from a different data source or system such as a lane closure system or a work zone management system. Fine-tuning the algorithm may improve classification accuracy for cases in types 2 and 3. Unfortunately, no good solutions are available for cases in type 4, but such cases rarely occur. Nevertheless, the discussion underscores the importance of properly documenting the presence of a work zone or work zone activities in the crash narrative.

### **3.4 Case Study 2: Distracted and Inattentive Driving Crashes.**

Two of the main reasons behind car crashes are distraction and inattentiveness. In Wisconsin, USA alone, from 2017 to 2019 the percentage of crashes due to distraction and inattentiveness increased from 8.28% to 12.41% (by statistical analysis from crash reports). When a driver fails to pay sufficient attention to perform basic tasks for safe driving, the driver is called inattentive, and the driving is called inattentive driving. While there is only action or activity behind inattentive driving, distracted driving involves both action and a source of distraction.

The definition and categorization of driver distraction and driver inattention can be referenced from existing sources and research. Dewar et al. (2007) stated that “the essential distinction between inattention and distraction is that inattention is internal to the driver and non-compelling, whereas distraction is external to the driver and compelling”. Regan et al. (2008) stated that the absence (in the case of driver inattention) of a competing activity is the key factor in differentiating driver distraction from driver inattention. Hoel et al. (2010) distinguished driver inattention from driver distraction according to the nature of the competing activity. For driver distraction this is any external non-driving-related activity and for inattention, this activity is preoccupation in internalized thought. Regan et al. (2011) defined distracted and inattentive driving, found relations between them, and made a taxonomy for them. They defined driver inattention as “insufficient or no attention to activities critical for safe driving” and categorized driver distraction as another form of driver inattention. According to the National Highway Traffic Safety Administration (NHTSA), distracted driving is defined as “Distracted driving is any activity that diverts attention from driving, including talking or texting on your phone, eating and drinking, talking to people in your vehicle, fiddling with the stereo, entertainment or navigation system — anything that takes your attention away from the task of safe driving” (Distracted Driving, NHTSA). In summary, driver distraction involves a triggering event, a competing activity; where the competing activity is externally generated and may lead to attention shift.

Though distracted driving is considered as a specific type of inattentive driving (Overview of the NHTSA’s Driver Distraction Program), the growing crash reports due to distraction lead us to consider them separately. Distracted driving involves some internal (i.e.,

inside the vehicle, for e.g., phone, radio, gps, etc.) or external sources, where inattentive driving does not involve any sources. It is important to differentiate the two types of crashes because knowing the source of distraction can help us take appropriate intervention. Furthermore, specific safety treatments for distracted or inattentive driving related crashes can be implemented for improved effectiveness.

### 3.4.1 Data Collection

In this case study, crash reports were acquired from the Wisconsin Department of Transportation (WisDOT) through the WisTransPortal data hub, including all the crash narratives. The data was collected during a transitional period of the database and observed several changes in the data elements, which are described below.

In the dataset before 2019, the field "DISFLAG" referred to all distracted and inattentive driving crashes (DOI). Therefore, the narratives that marked under DISFLAG flag could not describe which one was distracted and which one was inattentive narrative. In 2018, new data elements were added to the database to help separate distracted driving from inattentive driving. The implementation was rolled out gradually as law enforcement agencies upgraded their computer systems. Therefore, the database was not complete during that transitional time. In 2020, data was collected for the year 2017 to 2019 and found that there were 32,050 DOI crashes. In order to prepare a training dataset for DD and ID cases, some DOI cases were manually annotated as DD, ID, DD+ID, and ND cases. However, manually separating DD and ID from the huge DOI data set is not an effective method. It is expected that data collected in

a later time after the crash data improvement project such as 2019 to 2021 data are far better in distinguishing DD and ID from DOI with specific elements. That is why, after having a complete dataset for the years 2019 to 2020 and a partially complete dataset for the year of 2021 (till June 16), the distracted and inattentive crashes are populated based on the following query.

- *Distracted Crash* when {DISTACT [1,2] are not “Not Distracted” and not blank} & {DISTSRC [1,2] is not “Not Applicable”} (Not Distracted) and not blank} then it is Distracted crash, otherwise not
- *Inattentive Crash* when {ID} in DRVRPC [1,2] [A, B, C, D] then it is Inattentive crash, otherwise not

The field “DISTACT” provides the actions of drivers such as talking, listening, manipulating or other actions. The field “DISTSRC” provides the source of distraction such as hands-free mobile phone, hand-held mobile phone, vehicle-integrated device, or other source of distractions. The “DRVRPC” provides both inattentive driving and distracted driving parameters.

Table 3.4-1 presents the overall crash statistics after populating DD and ID crashes from 2019 to 2021. There were 51,405 DD cases and 17,791 ID cases in 2019-2020 dataset. From the experience of work-zone crash classification, it can be said that both datasets represent a good amount of data for using as training dataset in Noisy-OR. Therefore, individual classifiers for both DD and ID cases were developed.

**Table 3.4-1 Crash Statistics**

<b>Year</b>	<b>Type</b>	<b>Distracted (%)</b>	<b>Inattentive (%)</b>
2019	Reported	27,135 (21.16%)	9,994 (7.79%)
	Not Reported	101,074 (78.84%)	118,515 (92.21%)
	Total	128,209 (100%)	128,209 (100%)
2020	Reported	24,270 (23.9%)	7,797 (7.68%)
	Not Reported	77,239 (76.1%)	93,712 (92.32%)
	Total	101,509 (100%)	101,509 (100%)
2021 (partial dataset)	Reported	10,905 (23.72%)	3,612 (7.86%)
	Not Reported	35,077 (76.28%)	42,370 (92.14%)
	Total	45,982 (100%)	45,982 (100%)

For the DOI classifier, the DOI cases can be prepared from the DD and ID narratives in two ways: (1) either distracted or inattentive cases + both distracted and inattentive cases (2) both distracted and inattentive cases. Since the dataset is very noisy like work zone cases, the former method (either distracted or inattentive cases) will add noisy narratives to the training set from DD and ID cases. On the other hand, the other method (both distracted and inattentive cases) will reduce the noisy narratives in the dataset because it is unlikely for a narrative without any DD or ID related words to be reported as a both inattentive and distracted case. Both methods were used; and the one that provided higher accuracy to classify DOI cases was chosen.

The data from 2019-2020 was used as training data for all the classifiers. However, the classifiers for the cascade classifiers (models) cannot be tuned because the training data is very noisy (the narratives do not contain any DD, ID or DOI related words). Alternately, a random sample of 500 narratives for DD and 500 narratives for ID were manually annotated from DOI for 2018-2019 to determine the optimal threshold values for the classifiers (the details are described later). The use of manually annotated data ensures that these crashes are properly classified.

The 2020 dataset was used as the preliminary test data to investigate how well the classifiers are trained. The top 100 results of each classifier were selected, and manually investigated to get deeper insight about the classifiers. The 2021 dataset was used as final test dataset for all the classifiers to investigate how well the classifiers performed in a new dataset.

### **3.4.2 Result and Analysis**

In this section, the unigrams, bigrams, and trigrams with the highest probability scores and thus strong indicative of a DOI, a DD, or an ID narrative are presented. Next, the U, U+B, and U+B+T approaches are compared, and the best one is recommended for DOI, DD and ID classification.

#### *3.4.2.1 Unigram, Bigram, and Trigram Probability Analysis*

Table 3.4-2 Table 3.4-2to Table 3.4-4 show the top 25 unigrams, bigrams, and trigrams with their corresponding probability, positive count, and negative count for DOI, DD and ID. In these tables, DOI/NDOI means word count in DOI vs. word count in NDOI; same for DD/NDI and ID/NID. All calculations were performed by the equation of weighted count probability (Equation 3). It is clear that the method successfully obtained the most relevant unigrams, bigrams, and trigrams for each narrative type.

Table 3.4-2 lists the most important words related to the DOI classifier that have the highest probability. For example, the words “inattentive” and “distracted” has a probability of

0.99 and 0.97, respectively. With the Equation of simple count probability, the two words have the probability of 0.81 and 0.69, respectively. Therefore, the equation of weighted count probability works well to extract the most critical unigrams/bigrams/trigrams and give them high weights (probability). Some common phrases from the DOI narratives are (based on the manual review): not looking on the road, not paying attention, inattentive driving, operating phone/radio/GPS, reaching for drink/dropped phone or object, adjusting radio/visor, etc. Table 3.4-2 shows a good reflection of these phrases in all unigrams, bigrams, and trigrams. With these high probability scores of essential words, a test narrative will be given a high NoisyOR score when these words are present and thus classifying DOI cases from NDOI cases.

**Table 3.4-2 Top 25 Unigrams, Bigrams and Trigrams for the DOI Classifier**

<u>Unigrams</u>	<u>Pro</u>	<u>DOI/N DOI</u>	<u>Bigrams</u>	<u>Pro</u>	<u>DOI/NDOI</u>	<u>Trigrams</u>	<u>Pro</u>	<u>DOI/ND OI</u>
<b>inattentive</b>	0.9913	1632/39 1	inattentive driving	0.9907	1491/351	for inattentive driving	0.9888	1266/298
<b>distracted</b>	0.9727	931/409	for inattentive	0.9890	1290/303	cited for inattentive	0.9726	662/137
<b>paying</b>	0.9501	601/323	looked down	0.9813	845/183	inattentive driving unit	0.9314	383/78
<b>cell</b>	0.9266	443/256	was distracted	0.9610	544/128	not paying attention	0.9311	412/167
<b>looking</b>	0.9222	1557/15 50	looked up	0.9581	599/250	citation for inattentive	0.9231	360/86
<b>gps</b>	0.8555	256/150	distracted by	0.9513	480/124	was looking at	0.9138	373/194
<b>phone</b>	0.8516	1557/22 61	down at	0.9496	527/228	when he looked	0.9109	408/278
<b>radio</b>	0.8263	355/468	paying attention	0.9472	561/295	he was looking	0.9085	379/235
<b>reached</b>	0.8244	267/282	not paying	0.9377	439/173	was distracted by	0.9081	319/60
<b>looked</b>	0.7987	2878/50 72	looking at	0.9375	590/398	looked down at	0.9049	310/41
<b>asleep</b>	0.7888	473/800	was looking	0.9370	918/757	was not paying	0.8895	299/130
<b>eyes</b>	0.7706	318/525	cell phone	0.9279	432/229	he looked down	0.8826	275/71
<b>dropped</b>	0.7649	195/222	his phone	0.9274	413/196	when she looked	0.8752	285/159
<b>reaching</b>	0.7362	159/154	looked away	0.9271	376/103	she was looking	0.8501	253/162
<b>grab</b>	0.7345	142/48	phone and	0.9160	391/220	looked down to	0.8477	227/40

<b>floor</b>	0.7075	149/181	looked back	0.9098	429/319	driving unit 1	0.8441	298/301
<b>notice</b>	0.7006	254/512	looking down	0.9056	317/79	looked away from	0.8413	220/35
<b>bottle</b>	0.6734	114/84	her phone	0.8994	318/133	inattentive driving and	0.8378	218/53
<b>texting</b>	0.6708	108/27	driving unit	0.8958	470/443	he looked up	0.8378	224/96
<b>talking</b>	0.6599	144/256	at his	0.8823	419/413	was looking down	0.8127	195/50
<b>watching</b>	0.6507	136/243	at her	0.8696	349/330	down at his	0.8093	191/34
<b>attention</b>	0.6435	968/252 2	he looked	0.8382	951/1431	she looked down	0.8054	188/35
<b>inattentivel y</b>	0.6239	86/22	eyes off	0.8330	213/50	paying attention and	0.7942	189/112
<b>cigarette</b>	0.6145	82/30	attention to	0.8320	226/133	looking down at	0.7911	177/32
<b>coffee</b>	0.6141	88/95	she looked	0.8277	687/1052	she looked up	0.7896	178/59

Table 3.4-3 shows top unigrams, bigrams, and trigrams related to the DD classifier. From the unigrams column, it can be seen that, except for some unigrams like inattentive, asleep, etc., all the words are a strong indicator of distracted driving like distracted, GPS, cell, radio, phone, reached, talking, dropped, grab, reaching, floor, bottle, etc. Though “distracted” has the highest probability, given that police officers do not differentiate between distracted driving and inattentive driving on their written narratives, the word “inattentive” has the second-highest probability. From the manual review, it was frequently noticed that even when a narrative presents a distracted driving case, the last line of the narrative is something like “unit # is cited for inattentive driving.”. The frequent appearance of “inattentive” makes it very difficult to train the stable DD classifier. The same analysis is true for the bigrams and trigrams column of Table 3.4-2.

**Table 3.4-3 Top 25 Unigrams, Bigrams and Trigrams for the DD Classifier**

<u>Unigrams</u>	<u>Pro</u>	<u>DD/NDD</u>	<u>Bigrams</u>	<u>Pro</u>	<u>DD/NDD</u>	<u>Trigrams</u>	<u>Pro</u>	<u>DD/NDD</u>
<b>distracted</b>	0.9659	1906/523	looked down	0.9677	1450/266	for inattentive driving	0.9542	1912/662
<b>inattentive</b>	0.9560	2495/907	inattentive driving	0.9583	2284/786	cited for inattentive	0.9222	987/349
<b>paying</b>	0.9107	1071/479	for inattentive	0.9546	1953/678	was distracted by	0.8920	628/94
<b>looking</b>	0.8898	3025/1943	was distracted	0.9511	1080/197	not paying attention	0.8830	708/275
<b>cell</b>	0.8875	756/309	distracted by	0.9410	953/174	was looking at	0.8772	673/258
<b>phone</b>	0.8453	3247/2559	looked up	0.9329	1061/340	when he looked	0.8704	741/363
<b>gps</b>	0.8448	529/181	down at	0.9136	861/286	inattentive driving unit	0.8642	586/192
<b>reached</b>	0.8132	556/330	looking at	0.9102	1089/495	citation for inattentive	0.8525	548/181
<b>radio</b>	0.8116	713/522	paying attention	0.9075	1004/443	he was looking	0.8505	646/328
<b>looked</b>	0.7990	6103/5689	looked away	0.9029	696/147	looked down at	0.8497	498/73
<b>eyes</b>	0.7851	726/609	his phone	0.9017	767/252	was not paying	0.8352	529/220
<b>talking</b>	0.7500	413/280	was looking	0.8982	1725/1006	when she looked	0.8284	507/208
<b>dropped</b>	0.7391	386/259	not paying	0.8902	748/288	he looked down	0.8211	446/104
<b>grab</b>	0.7193	289/66	cell phone	0.8876	731/281	she was looking	0.8069	472/222
<b>attention</b>	0.7122	2371/2788	phone and	0.8858	711/266	he looked up	0.7842	393/135
<b>reaching</b>	0.7113	313/179	looked back	0.8827	822/396	driving unit 1	0.7813	558/417
<b>asleep</b>	0.7053	1057/1231	her phone	0.8565	553/171	looked down to	0.7812	368/57
<b>notice</b>	0.6960	547/594	looking down	0.8558	530/126	looked away from	0.7763	361/58
<b>floor</b>	0.6876	296/202	at his	0.8420	769/490	she looked down	0.7521	327/52
<b>ejected</b>	0.6813	366/350	at her	0.8307	639/383	paying attention and	0.7462	352/169
<b>owi</b>	0.6646	1382/1804	she looked	0.8234	1433/1178	inattentive driving and	0.7460	334/119
<b>alcohol</b>	0.6638	507/603	driving unit	0.8222	821/605	was looking down	0.7443	323/85
<b>bottle</b>	0.6547	228/104	he looked	0.8189	1920/1640	he looked back	0.7300	331/168
<b>seat</b>	0.6512	1927/2611	eyes off	0.7818	375/87	she looked up	0.7245	299/86
<b>def</b>	0.6452	724/959	attention to	0.7768	400/180	down at his	0.7229	292/58

Table 3.4-4 shows top unigrams, bigrams, and trigrams related to the ID classifier. From the unigrams column of Table 3.4-4, it can be seen the reflection of the previous statement. The same observation is valid for the bigrams and trigrams column of Table 3.4-4. One important

observation is that the highest probability for the unigram and bigram “inattentive” is 0.9642 and 0.9623, respectively; which is higher than that in Table 3.4-3. This indicates that DD classifier is a less effective classifier to separate DD cases from ID cases with the presence of ID classifier.

**Table 3.4-4 Top 25 Unigrams, Bigrams, and Trigrams for the ID Classifier**

<u>Unigrams</u>	<u>Pro</u>	<u>ID/NID</u>	<u>Bigrams</u>	<u>Pro</u>	<u>ID/NID</u>	<u>Trigrams</u>	<u>Pro</u>	<u>ID/NID</u>
<b>inattentive</b>	0.9642	2148/1254	for inattentive	0.9623	1665/966	for inattentive driving	0.9622	1630/944
<b>paying</b>	0.8816	757/793	inattentive driving	0.9623	1926/1144	cited for inattentive	0.9531	874/462
<b>distracted</b>	0.8435	1045/1384	looked down	0.9180	928/788	inattentive driving unit	0.9176	498/254
<b>cell</b>	0.8423	496/569	not paying	0.8894	554/482	citation for inattentive	0.9003	455/274
<b>looking</b>	0.8070	1950/3019	down at	0.8811	585/562	not paying attention	0.8819	520/463
<b>asleep</b>	0.8022	904/1384	paying attention	0.8798	709/738	looked down at	0.8558	342/229
<b>gps</b>	0.7400	287/423	looked up	0.8797	689/712	he was looking	0.8504	472/502
<b>reached</b>	0.7020	315/571	was distracted	0.8660	613/664	was not paying	0.8481	389/360
<b>phone</b>	0.6902	1855/3951	looking down	0.8529	364/292	was looking at	0.8340	437/494
<b>radio</b>	0.6860	409/826	distracted by	0.8503	530/597	inattentive driving and	0.8286	284/169
<b>dropped</b>	0.6755	232/413	cell phone	0.8486	484/528	he looked down	0.8286	308/242
<b>grab</b>	0.6699	160/195	driving unit	0.8470	634/770	when he looked	0.8278	493/611
<b>reaching</b>	0.6569	184/308	looked away	0.8468	420/423	was distracted by	0.8215	353/369
<b>watching</b>	0.6436	198/376	was looking	0.8425	1167/1564	driving unit 1	0.8113	420/530
<b>looked</b>	0.6432	3495/8297	looking at	0.8347	687/899	when she looked	0.8025	334/381
<b>texting</b>	0.6426	126/104	his phone	0.8341	469/550	looked away from	0.7885	243/176
<b>eyes</b>	0.6347	402/933	her phone	0.8240	356/368	looked down to	0.7881	244/181
<b>bottle</b>	0.6242	134/198	phone and	0.8187	437/540	she was looking	0.7843	313/381
<b>floor</b>	0.6230	170/328	fell asleep	0.8032	689/1024	he looked up	0.7803	263/265
<b>notice</b>	0.6162	336/805	looked back	0.8025	506/712	was looking down	0.7736	230/178
<b>inattentively</b>	0.6158	104/42	eyes off	0.7850	250/212	down at his	0.7677	215/135
<b>tired</b>	0.6034	160/332	at his	0.7791	496/763	he fell asleep	0.7641	304/409
<b>cigarette</b>	0.5946	97/93	attention to	0.7747	273/307	paying attention and	0.7578	246/275
<b>adjust</b>	0.5812	92/111	at her	0.7641	402/620	looking down at	0.7445	195/126
<b>coffee</b>	0.5808	102/164	he fell	0.7532	310/448	she looked down	0.7441	205/174

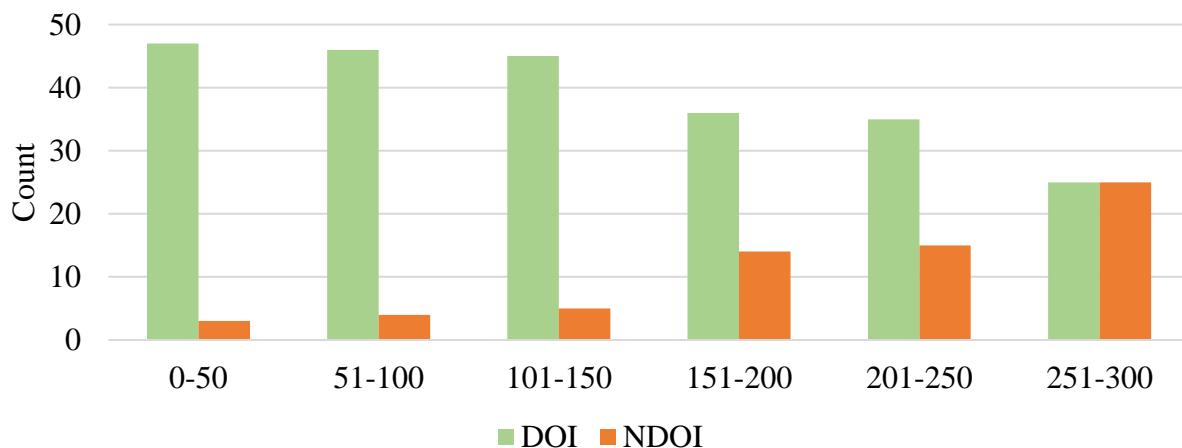
### 3.4.2.2 DOI, DD, and ID Classifiers

In this section, distracted and inattentive versus neither classification using DOI classifier, *distracted* versus *neither* classification using DD classifier, and inattentive versus neither -classification using ID classifier are performed. A threshold value for probability scores was used to exclude unigrams, bigrams, and trigrams with low probability scores. For example, if a threshold value of 0.50 is set in the U+B approach, then all the unigrams and bigrams having probability scores (by Equation 1 or 3) less than 0.50 are not considered. The best threshold value for each classifier was determined by searching in the range of 0.00 to 0.90 with 0.05 increments. Three metrics; Accuracy, AUC and ROC, were used to find the best threshold. The accuracy value is the ratio of the number of narratives that are correctly classified divided by the total number of narratives. The ROC is a graph that shows the performance of a classification model over the entire range of sensitivity and specificity, and AUC measures the area underneath the ROC curve. Among the three, Accuracy was found to be the best metric. Therefore, the Accuracy value was used as the evaluation metric for comparing the performances of the classifiers.

For the DOI classification, when tested on the 300 manually reviewed validation data set described earlier, among U, U+B, and U+B+T, the U+B approach (by both simple Count and weighted count probability equations) performs the best, and U+B+T performs the worst. The U+B+T contains trigrams, and three consecutive words are generally rare, and those found in the training data are unlikely to repeat in the test narratives. For example, “inattentive in his” is a trained trigram with 0.93 probability, but the possibility of these exact three words appearing in a test narrative in the same order is low. The test narrative may have trigrams like “inattentive in

her”, “inattentive on his”, “inattentive because his”. For this reason, the U+B+T approach does not work well as expected, but with a large enough training dataset, it may perform better. With the equation of simple count probability, the best result was achieved using the U+B approach with a 0.35 cutoff value. With the equation of weighted count probability, the best result was achieved using the U+B approach with a 0.5 cutoff value (Figure 3.4-1).

Next, the U+B approach using both equations were applied to the 2020 and 2021 NDOI test set. The top 100 results of 2020 and the top 300 results of 2021 were manually reviewed and verified. The breakdown of the top 300 manual reviews of 2021 using weighted count probability is shown in Figure 3.4-1. The accuracy of classifying DOI cases from NDOI cases is 78% using the weighted count probability. Figure 3.4-1 shows that the DOI cases have a consistent rate in the top 300 results.

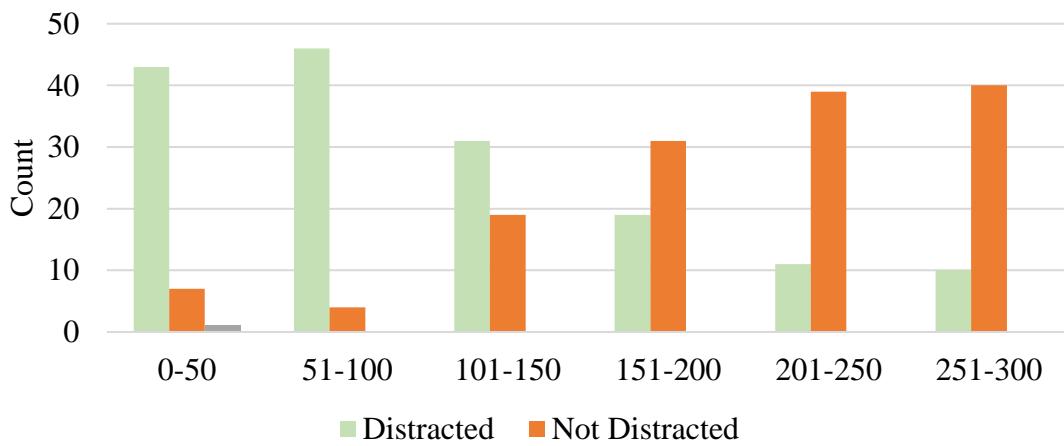


**Figure 3.4-1 DOI Classifier Using the U+B Approach (Eq. 3, 0.50 Cutoff Threshold)**

From the Accuracy values obtained on the 500 manually reviewed validation set (described earlier) for the DD classifier, among U, U+B, and U+B+T, U+B approach for both

simple count probability and weighted count probability perform the best and U+B+T performs the worst. The reason behind that, as was for the DOI classifier, is that U+B+T contains trigrams, and three consecutive words found in training data are unlikely to be repeated in the test narratives. With the equation of simple count probability, the best result was achieved using the U+B approach having a 0.75 cutoff value. With the equation of weighted count probability, the best result was achieved using the U+B approach having a 0.65 cutoff value.

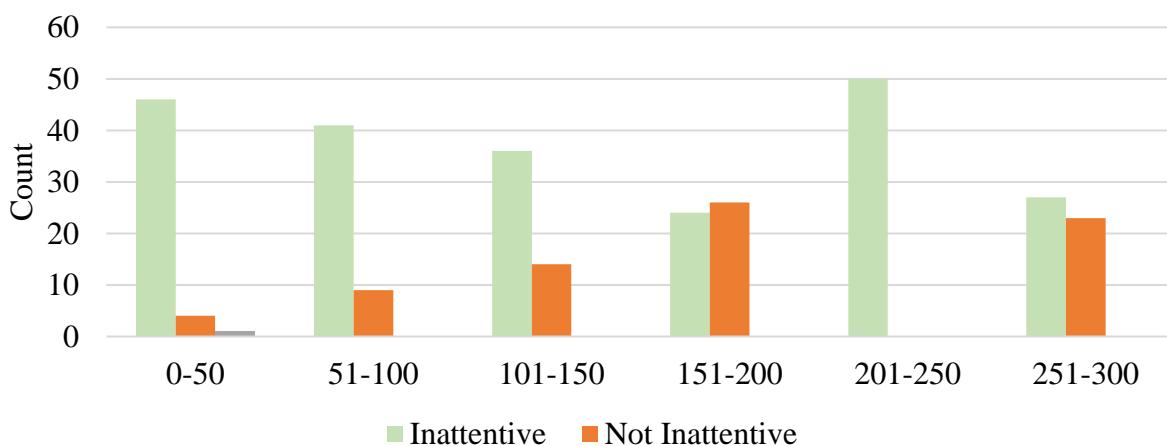
Next, this U+B approach using both equations are applied to the 2020 and 2021 NDD dataset and the top 100 results of 2020 and 300 result of 2021 were manually reviewed and verified. The result's breakdown of the top 300 manual reviews using weighted count probability is shown in Figure 3.4-2. From Figure 3.4-2, the equation of weighted count probability is good at finding distracted cases in the top 300 results of the DD classifier. The accuracy of classifying DD cases from NDD cases is 53.33%. It also shows a consistent rate of DD cases in the top results.



**Figure 3.4-2 DD Classifier Using the U+B Approach (Eq. 3, 0.65 Cutoff Threshold)**

From the Accuracy values obtained on the 500 manually reviewed validation set of the ID classifier, among Unigram, U+B, and U+B+T, U (by simple count probability) and U+B (by weighted count probability) approach perform the best, and U+B+T perform the worst. The reason is the same as was for DOI and DD classifiers, trigrams are rare. With the equation of simple count probability, the best result was achieved using the U approach having a 0.45 cutoff value. With the equation of weighted count probability, the best result was achieved using the U+B approach having a 0.75 cutoff value.

Next, these U and U+B approaches using both equations are applied to the 2020 and 2021 NID dataset and the top 100 results of 2020 and top 300 result of 2021 were manually reviewed and verified. The result's breakdown of the top 300 manual reviews using weighted count probability is shown in Figure 3.4-3. From this figure, the equation of weighted count probability is good at finding inattentive cases in the top 300 results of the ID classifier. The accuracy of classifying ID cases from NID cases in 2021 dataset is 63.33%. However, there is no consistent rate of ID cases in the top results.



**Figure 3.4-3 ID Classifier Using the U+B Approach (Eq. 3, 0.75 Cutoff Threshold)**

The summary of all three classifier's best results using both the equation of simple count probability and the equation of weighted count probability is shown in Table 3.4-5. As discussed earlier, the U+B approach yields better results most of the time. Only in one case, U approach shows more promising results. It can also be seen that the DOI classifier yields very high accuracy values, followed by the ID classifier and finally, by the DD classifier. In general, the DOI classifier performs the best to separate DOI cases from NDOI cases, where the DD classifier performs the worst to separate DD cases from NDD cases. On the other hand, the ID classifier shows satisfactory performance. In the top results, the ID classifier offers the best performance, followed by the DOI classifier and finally, by the DD classifier. In general, the DD classifier performs the worst in every sector, which also degrades the overall performance on the DD versus ID classification. As a cascaded approach is used to classify cases, every classifier needs to perform well to achieve an overall good result.

**Table 3.4-5 Summary of All Results of Best Approaches**

Classifier	Best Approach	Equation	Threshold	Accuracy (Top 100 results of random sample from 2018-2019 dataset to find the best threshold)	Top 100 results of entire-2020 dataset	% of top 300 results of entire-2021 dataset
DOI	U+B	Simple count probability	0.35	78	89	91.00
	U+B	Weighted count probability	0.5	78	91	78.00
DD	U+B	Simple count probability	0.75	65	84	51.00
	U+B	Weighted count probability	0.65	54	57	53.33
ID	Unigram	Simple count	0.45			82.33

	probability		45	70	
U+B	Weighted count probability	0.75	49	98	63.33

### 3.4.3 Summary of Observations

For this case study, various forms of words with NoisyOR were applied to identify DOI, DD, and ID crashes from crash narratives. 2018 and 2020 dataset were used as training dataset, and 2021 as test dataset because of the updates to the distraction data field in 2018. Those methods were based on probability scores of unigrams, bigrams, and trigrams and combining them using Noisy-OR. A new, improved way of computing probability scores was introduced to suit the task. The method worked on automatically generated training data that required no manual effort. The classifiers obtained good results despite the noise in the training data.

Overall, the DOI classifier with simple count probability method worked well in a new (2021) dataset compared to the DOI classifier with weighted count probability. The threshold value is 0.35 for simple count probability and 0.5 for weighted count probability. Since a lower threshold value means keeping more keywords for the classifier, the DOI classifier with simple count probability searches more keywords in the narratives compared to that in the classifier of weighted count probability. A DOI narrative that does not have strong DOI related words (high probability scores) can be handled well by the simple count probability method.

The performance of the DD classifier with weighted count probability is consistent in both known (2018-2020) and new (2021) dataset compared to that of the DD classifier with

simple count probability. The threshold value of the DD classifier with weighted count probability is lower than that of the DD classifier with simple count probability. Therefore, similar conclusion can be drawn for the DD classifier as the conclusion was drawn for the DOI classifier with simple count probability. The higher threshold value of 0.65 (compared to that of the DOI classifier) indicates that the DD classifier with weighted count probability will work well in the narrative that has strong DD related keywords.

The performance of the ID classifier with simple or weighted count probability is not consistent. In the random sample dataset, the accuracy is very low, indicating that the ID classifier does not perform well in the narratives that do not contain strong ID related keywords. The ID classifier with simple count probability performs well in finding ID cases in the top results of known (e.g., training) and new (e.g., test) dataset. Though the classifier with weighted count probability performed well in the known dataset, it did not perform well in the new dataset. The possible reason is that the ID dataset is very noisy. During the manual review process, it was found that most of the time ID narratives did not contain any inattentive related keywords. Moreover, most of the keywords in the inattentive cases were related to distracted driving, indicating that attentive driving cases were not properly recorded in the structured data.

In summary, the performance difference for the DD and ID classifiers in known and new dataset shows that the new test dataset has either many keywords that are not present in the training dataset or the keywords that are not strongly related to DD and ID cases.

### **3.5 Conclusion and Recommendations**

In this research, keyword-based text classifiers were developed using NoisyOR to identify misclassified crashes from the crash narratives of police reports. Specifically, the unigram +bigram NoisyOR classifier was developed and proven to be an effective means to recover WZ crashes from those police officers did not flag correctly. The narrative of a crash may not contain any relevant information linking it to that crash, and narratives that are not flagged to a specific crash type may contain information related to that crash. The NoisyOR method was used because of its ability to work effectively despite the high level of noise in the unstructured text or crash narratives. Moreover, NoisyOR does not require much training time, is computationally efficient and is easier to implement. The method was tested using two case studies: work-zone crashes and distracted and inattentive crashes.

For Work-zone case study, a manual review of the top 450 cases classified as WZ crashes in the testing data recovered 201 missed WZ crashes, which was 0.24 % of the testing data. The review also indicated that beyond 450 cases, the chance of having missed WZ crashes may be very low. A follow-up analysis revealed that 73.13 % of the missed crashes occurred from 8 a.m. to 6 p.m., with a high percentage happening from 4 p.m. to 5 p.m. A large percentage of those crashes occurred in the summer (July and August) and 43 % occurred on urban city streets. The narratives of the cases that have high NoisyOR scores but are not WZ crashes were carefully reviewed and categorized into the five following groups:

- 1) Cases with positive words for location or address such as “the Zoo”, “Zoo interchange”: This issue is caused primarily by major roadway construction projects that span over multiple years, multiple stages and phases and multiple areas.

- 2) Cases with positive words for (temporal) traffic control devices such as “concrete barrier”, “median cement”, “attenuator” and “barriers”: Many of these devices, such as median concrete barriers, are permanently deployed to channelize traffic or to protect overpass and underpass structures such as an attenuator near a bridge or at a gore area.
- 3) Cases with weak positive words for traffic situations such as “congestion” or “backup” which are caused by non-WZ events (i.e., regular congestion or secondary crashes).
- 4) Cases with strong positive words such as “orange construction” or even “construction zone” whose situations are actually not related to a work zone location or work zone activities.
- 5) Undecided cases, even after a manual review: The authors were conservative and categorized undecided crashes from this study as NWZ crashes.

A location and/or time that a work zone crash occurred can certainly improve WZ classification in types 1 and 5. Such information, however, has to be linked to and retrieved from a different data source or system such as a lane closure system or a work zone management system. Fine-tuning the algorithm may improve classification accuracy for cases in types 2 and 3. Unfortunately, no good solutions are available for cases in type 4, but such cases rarely occur. Nevertheless, the discussion underscores the importance of properly documenting the presence of a work zone or work zone activities in the crash narrative.

The second case study was conducted to find misclassified DOI, DD and ID crashes from the narratives that are reported as NDOI, NDD and NID, respectively. Various word forms, such as unigrams, bigrams, and trigrams, as well as several threshold values for word probability and

classification score were used to develop several NoisyOR classifiers. In addition to the previous word probability function, a weighted count word probability function was constructed and evaluated. Overall, the performance of the DOI classifier with simple count probability was satisfactory. In contrast, the performance of the DD classifier with weighted count probability was consistent, but not as satisfactory as DOI. The performance of the ID classifier using either simple count probability or weighted count probability was inconsistent.

Compared to the work zone classifier that does not require any threshold value, the DD and ID classifiers perform poorly even in the presence of optimal threshold. The good performance of the DOI classifier helps us understand the reasons behind it. The data of work zone crashes is more mature in the structured dataset than DD and ID. Even though the DD and ID crashes are well defined in the data manual, the persons who record the data (e.g., police officers) may not be well familiar with the new data fields in the structured data or have difficulty distinguishing DD and ID crashes.

Based on the lessons learned of this study, the following recommendations have been suggested:

1. The NoisyOR can be used for crash classification from imbalanced and noisy dataset of text narratives.
2. NoisyOR will be a best option if most of the narratives in the dataset are lengthy narratives ( e.g, length more than 200 words).

3. The accuracy of NoisyOR is consistant, which helps to formulate a regression model. The regression model can be used to determine the optimam or near optimum number of narratives to review if required.
4. In case a narrative carries information related to two crash types (e.g., distracted and inattentive), their individual NoisyOR classifiers can be used to separate them.
5. NoisyOR is very simple and theoritically sound that requires less comoputational power.

## **Chapter 4. METHODOLOGY COMPARISON**

### **4.1 Introduction**

Besides regular road maintenance activities, highway construction projects are on the rise due to aging road infrastructure, activities lead by economic growth, and increasing demands for freight and passenger transportation. Construction projects disrupt traffic flow, create new traffic patterns, and cause congestion at or near work site or work zone (WZ). Moreover, WZs can become major highway hotspots of crashes that cause injuries and deaths to drivers, passengers, vulnerable road users, and construction workers. On average, 773 people died every year from 1982 to 2017 due to construction and maintenance work on the roads (CDC, 2020). In Wisconsin, more than 3,100 crashes occurred in WZ or WZ impact areas in 2019, including 899 injuries and 18 deaths. The WZ crash statistics, however, may be an understatement of the actual safety problem. Research shows that police officers miss to label many WZ crashes due to a variety of reasons, such as design deficiencies in crash reporting form (Blackman et al., 2020; Ullman & Scriba, 2004; J. Wang et al., 1996), misjudgment (Farmer, 2003; J. Wang et al., 1996), ignorance of minor or no injuries (J. Wang et al., 1996), and lack of proper knowledge about WZ crashes (Graham & Migletz, 1983). Incomplete crash statistics will undermine the planning, design, and implementation of roadway improvement projects. Hence, it is necessary to recover the missed WZ crashes.

Police officers report crashes in two data formats: structured data in a tabular format and unstructured data in a text format. Structured data are composed of a fixed number of columns,

which limit the details of crash events. Unstructured data, also called crash narratives, include explanation of the possible causes of the crash, witness testimony, and other important information. Research shows that crash narratives carry a large amount of valuable information which is not normally recorded in structured data fields (e.g., the type and location of construction equipment and materials, the visibility and understandability of construction signs to drivers, and the situation where lanes are closed or merged). The information in crash narratives that is highly related to construction activities can help in identifying missed WZ crashes. Manually checking the crash narrative is a common way to identify missed WZ crashes, but this process is very labor intensive and time consuming. In this study, several state-of-the-practice machine learning (ML) techniques were explored and evaluated to detect missed WZ crashes from crash narratives. Proven to be competent and successful in other domains, the natural language processing (NLP) combined with ML can extract essential information, remove redundancies, and automatically learn patterns from unstructured text data to classify a narrative.

## 4.2 Methodologies

This section will discuss the principles and procedures that will guide the study of crash narratives. To reduce manual work and speed up the process of mining crash narratives, several machine learning techniques are considered in addition to probabilistic. This study implemented (1) multinomial naive bayes (MNB), (2) logistic regression (LGR), (3) support vector machine (SVM), (4) random forest (RF), (5) K-nearest neighbor (K-NN), (6) recurrent neural network with Gated Recurrent Unit (GRU), (7) Hierarchical Attention based neural Network, and (8) a probabilistic model that uses NoisyOR to combine probabilities. The NoisyOR and the

Hierarchical Attention based neural network method have never been used in research on highway safety. Each model requires annotated training data (crash narratives); one serves as a positive sample, while the other serves as a negative sample.

#### **4.2.1 Multinomial Naive Bayes (MNB)**

MNB is a classical text mining technique that is used in document classification. In MNB, the narrative is treated as a set of words. It uses a fixed set of words to define input vector, where the values in the vectors represent word frequencies in the narratives. The probability that a narrative indicates WZ crash is calculated by combining the prior probability of a narrative to be in WZ class with the conditional probabilities of words given that a narrative is in WZ class. The conditional probabilities are estimated by a smoothed version of maximum likelihood estimation that uses relative frequency counting. The Laplace smoothing was applied to calculate relative frequency. More details on MNB can be found in (Manning et al., 2008).

#### **4.2.2 Logistic Regression (LGR)**

LGR is a supervised linear classification algorithm which models the narratives using a logistic function called sigmoid function (Kantardzic, 2011). It takes real numbers (i.e. features) as input and provides outputs between 0 and 1; and predicts the odds of being a narrative WZ based on the values of the independent variables. In this study, the independent variable

represents the weight (i.e., count frequency, tf-idf) of the words. L2 penalization was applied in objective function to handle multicollinearity and overfitting problems (X. Zhang et al., 2019).

#### **4.2.3 Support Vector Machine (SVM)**

SVM was developed by Vapnik and his colleagues (Boser et al., 1992; Cortes & Vapnik, 1995) based on the principle of structural risk minimization in statistical learning theory. SVM has been found to be effective in many text classification problems such as hazard analysis (Zhong et al., 2020), news article categorization and sentiment prediction (e.g., Joachims, 1998; Pang et al., 2002). It has been claimed to be less prone to overfitting (Joachims, 1998). As a supervised learning method that can be used for regression and classification, SVM uses kernels to map data in low dimensional space to a higher dimensional feature space and generates a hyperplane to separate the data by class. In this study, words were used as features and their number of occurrences as feature values. The linear classifier kernel was used because of its common use in text mining given that the word-based feature space is already very high dimensional.

#### **4.2.4 Random Forest (RF)**

Random Forest was first developed by Tin Kam Ho (Ho, 1995) based the random subspace method. RF is a learning algorithm based on ensembles of decision trees. RF is very popular in the field of pattern recognition and machine learning, and is used to solve high-

dimensional classification problems (LEO Breiman, 2001). It fits several decision tree classifiers on various randomly selected sub samples (drawn with replacement) from the training set and takes the average of all probability predictions of the trees to improve accuracy and control overfitting (Leo Breiman, 1999).

#### **4.2.5 K-Nearest Neighbor (K-NN)**

K-NN is a non-parametric lazy machine learning algorithm used in the field of pattern recognition. It is one of the most widely used data mining techniques in classification problems. The classification score is calculated based on the majority votes of the k nearest narratives where the nearness is computed using a suitable distance measure. In this study, there are only two classes - WZ and Non-WZ or NWZ. The performance of K-NN depends on the distance measure used. Therefore, an appropriate distance measure must be selected to achieve the best K-NN performance. The two most common distance measures in the text classification field are Euclidean distance and cosine distance. Euclidian distance metric was applied, which is an extension of Pythagoras's theorem in multi-dimensional space. It calculates the distance by taking the square root of the sum of the squares of the difference between two narrative vectors and the value ranges from 0 to any positive number. Different values of K (i.e. 3, 5, 7, 9) were tested for K-NN and it was found that K =7 gave the best result. More details of the K-NN can be found in (Cunningham & Delany, 2020).

#### **4.2.6 Gated Recurrent Unit (GRU)**

Recurrent neural network (RNN) is a special neural network architecture for handling sequential data such as text narratives which are sequences of words. While processing sequential inputs one item at a time (for example, one word at a time), RNN needs a mechanism to learn to remember important items it saw earlier and forget the unimportant items. This is achieved using special neural networks cells. The two most used such cells are Gated Recurrent Unit (GRU) and long short-term memory (LSTM). The GRU proposed by Cho et al. (Cho et al., 2014) is similar to LSTM with a forget gate (Gers & Cummins, 1999); but compared with traditional LSTM, it has a shorter training time and fewer parameters. Due to these advantages, GRU was used in this work. The GRU does not have any cell state like LSTM and uses hidden state to transfer information. The GRU has two gates: the reset gate decides how much past information will be transferred to the next step, and the update gate decides what information to be added or discarded to the current layer. The update gate is very similar to the forget and input gate of an LSTM. Readers are referred to (Cho et al., 2014) for more information.

### **4.3 Data Collection and Pre-Processing**

In this study, the previous work-zone dataset was used that are discussed in chapter 3. Research shows that machine learning algorithms cannot provide good performance for an imbalanced dataset that has far fewer number of examples of one class compared to the other (Jeong et al., 2018). Based on the ratio of reported WZ and NWZ crash narratives, the data set can be called an unbalanced dataset (Leevy et al., 2018). A balanced dataset can be obtained by

oversampling or undersampling. For some classifiers (such as SVM), oversampling can degrade their performances (Glen, 2019). To create a balanced dataset for MNB, LGR, SVM, RF, K-NN and GRU, 2000 WZ crash narratives and 2000 NWZ crash narratives were randomly selected from crash reports from the years 2017 to 2018, resulting in the number of feature vectors in training set to be 4,000. Similarly, another 4,000 were chosen as validation data from crash reports from the years 2017 to 2018. The crash narratives from the year 2019 were used for evaluation.

Simple data processing techniques, such as case folding (turning words into lowercase), punctuation and word spacing removal techniques are applied to prepare data for all methods. In addition, all redundant terms (such as stop words,) and the words with length (number of characters) less than 4 were deleted from the narratives for the methods GRU, LGR, SVM, RF, and K-NN models.

#### **4.4 Feature Generation and Model Parameters Tuning**

The tasks of feature generation and model parameter tuning are different among the classifiers, which are described sequentially in this subsection. In *Google Colab*<sup>6</sup>, the machine learning libraries *TensorFlow*<sup>7</sup> for GRU, and *sklearn* for MNB, LGR, SVM, K-NN and RF

---

<sup>6</sup> Google Colab or Colaboratory provides free GPU access to write and execute Python in web browser, and easy code sharing with others. <https://colab.research.google.com/notebooks/intro.ipynb#recent=true>

<sup>7</sup> TensorFlow provides prebuilt data processing and deep learning framework to develop models. [https://www.tensorflow.org/api\\_docs/python/tf/keras/layers/GRU](https://www.tensorflow.org/api_docs/python/tf/keras/layers/GRU)

were used to generate features and develop models. After processing the narratives, narratives were converted into tokens (unigrams) by count vectorization. After trying input vectors with various lengths such as 50, 100, 200, 300, 500, 1000, 5000, and the full length of vocabulary, it was found that the input vector with length 500 gives the best result for MNB, LGR, SVM, K-NN and RF in the reported WZ in the training dataset. In this process, a vocabulary was prepared that only considers the top 500 words ordered by word frequency across the narratives. Other model-specific parameters of all models were fine-tuned based on training dataset and used the best parameters for evaluation. Advanced data processing techniques such as lemmatization, bigram tokenization, tf-idf weighting, and different vectorization architectures were applied to train MNB, LGR, SVM, K-NN, and RF. No significant improvement was observed, rather the model performance degraded for some techniques.

For GRU, 154, the third standard deviation of the narrative length (154) was used as the input vector length and then post padding was applied to the vector to fill with 0 if the input length was shorter. The tokens were converted to vectors using pre-trained Google *word2vector*<sup>8</sup>. Words that did not exist in the dictionary were initialized with a random number in the range of 0 to  $\sqrt{0.25}$  using Gaussian distribution. The GRU was developed by stacking two GRU layers (each containing 32 hidden units) and a Dense layer (containing 1 hidden unit with sigmoid function). The Dense layer provides the final output of the model. The binary cross-entropy was used as the loss function, 32 as the mini-batch size, Adam as the optimizer, and the early stopping in the callback function to find the best model.

---

<sup>8</sup> Google wor2vec provides an implementation of the continuous bag-of-words and skip-gram architectures for computing vector representations of words that is used in many natural language processing applications.  
<https://code.google.com/archive/p/word2vec>

## **4.5 Evaluation Methodologies**

Two evaluations were conducted. The first evaluation compared classification performances of the classifiers on a fixed gold-standard test dataset of WZ and NWZ narratives. Although this is the standard type of evaluation for text classification and reflects how well a classifier can distinguish between WZ and NWZ narratives, it does not necessarily reflect how well a classifier may identify missed WZ crashes from a large number of NWZ narratives. Given that the latter task is the reason why these classifiers were built, a second evaluation was conducted specifically designed to measure the performance on identifying missed WZ crashes from NWZ narratives.

### **4.5.1 Evaluation of Classification Performance**

For the first evaluation, the standard evaluation metric of area under ROC curve (AUC) was used to measure overall performance of the classifiers. 100 WZ and 100 NWZ reported crash narratives were randomly selected from the 2019 crash reports, and manually marked the true positives (missed WZ crashes) and true negatives (true NWZ crashes). It was found that only 36 cases were truly positive in WZ narratives, and all were true negatives in NWZ narratives. The surprising result of WZ narratives forced us to review another 200 WZ reported cases, and that time 56 true positives were found. In this way, 92 true WZ cases and 208 NWZ cases were found from the 300 reported WZ narratives; and 100 true NWZ cases from the 100 reported NWZ narratives. In total, the test dataset contained 92 WZ cases and 308 NWZ cases out of the total 400 crash reports. This manually reviewed dataset was used to compare the

models using AUC. It can be inferred from the above numbers that the WZ crash narrative in the training data contains approximately 70% or 208/300 noisy narratives that do not contain any WZ related words. During the manual review process, it was observed that only a few important keywords or phrases in the true WZ narratives are relevant for classification.

#### **4.5.2 Evaluation for Identifying Missed WZ Crashes**

Although the AUC method helps in evaluating the classification performance of the classifiers, it does not help in evaluating the performance of the classifiers in identifying missed WZ crashes. Therefore, a second evaluation was conducted which closely reflects the classifier's ability to find missed WZ crashes from reported NWZ crash narratives. All the classifiers were tested on the 2019 NWZ narratives (total 82,215 crash reports that were flagged as NWZ by the "CONSZONE" flag) and collected the top 100 narratives of each classifier ranked by the classification scores assigned by the classifier. A manual evaluation was conducted on the top 100 narratives of each classifier. The classifier that includes the maximum number of missed WZ crashes in its top 100 narratives is deemed as the best classifier.

#### **4.6 Findings of WZ Related Words**

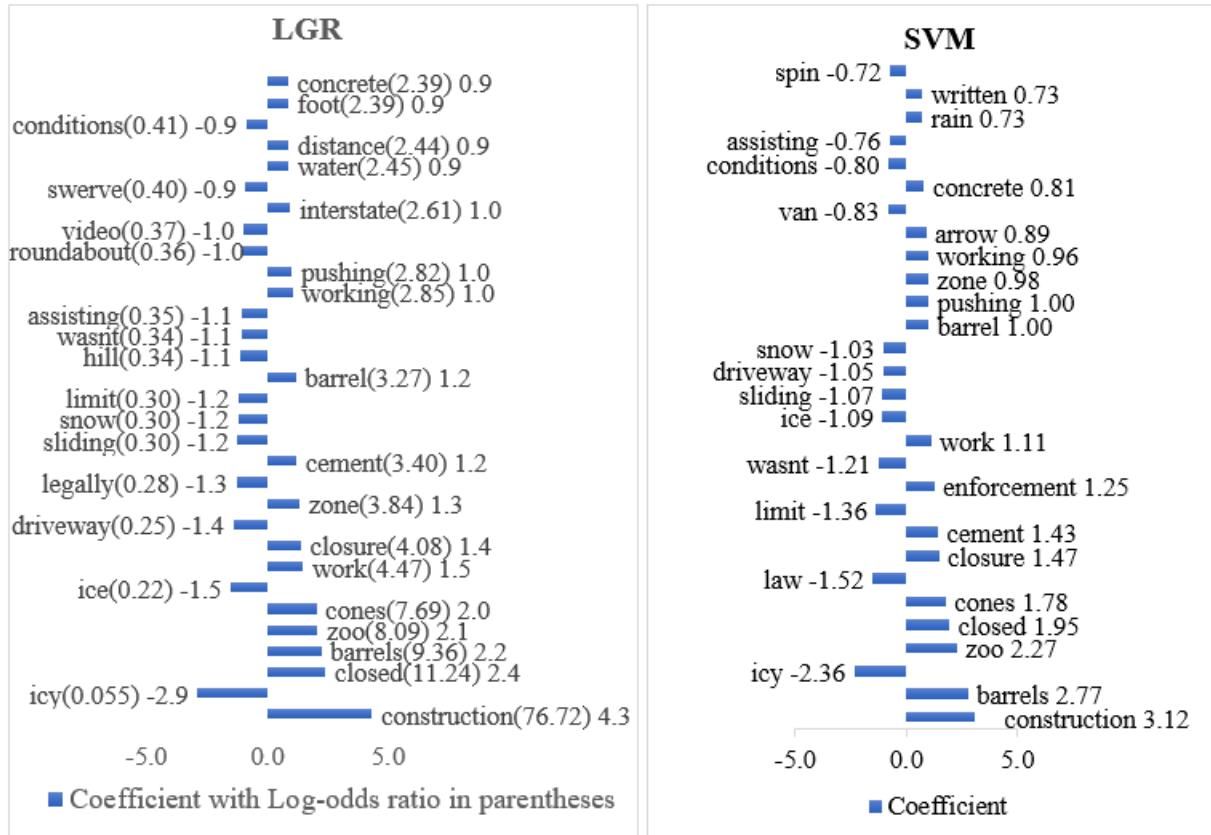
In this research, 7 classifiers: MNB, LGR, SVM, K-NN, RF, GRU, and Noisy-OR have been applied. During the manual review for evaluation purpose, it was observed that the narrative could have multiple WZ-related keywords that would help identify missed WZ crashes.

Therefore, analyzing the keywords captured by a classifier during the training phase can provide insights about the classifier.

The results of LGR (X. Zhang et al., 2019) and linear SVM (Chang & Lin, 2008; Cuingnet et al., 2011; Guyon et al., 2013) can be explained by the magnitudes of the coefficients of the words. However, their interpretations are different. In linear SVM, if the absolute value of the coefficient of the vector component (word) is smaller than the coefficients of other components, the coefficient of this component has little influence on the classification result, and vice versa (Cuingnet et al., 2011). In LGR, the coefficient indicates the log odds of being a positive class (WZ crash), and the exponent of coefficient ( $e^{coefficient}$ ) indicates the log odds ratio. For example, the log odds of WZ crash is 4.34 times higher when a narrative has the token “construction” (Figure 4.6-1) compared to the narratives that do not have this word. Or the odd of a narrative being a WZ crash is 76.71 times higher for the presence of word “construction” compared to not present of that word in the narrative. Due to difference in interpretations, the LGR cannot be directly compared with SVM using the coefficient of component (token).

Figure 4.6-1 shows the top 30 words of LGR and SVM. There are 14 words with positive coefficients and 16 words with negative coefficients in LGR, and there are 17 words with positive coefficients and 13 words with negative coefficients in SVM. LGR's log odds ratio and SVM's coefficients (in descending order) can provide potential insights into how these tokens affect the classifier. Positive coefficients indicate the direction of positive class (WZ crash) and negative coefficients indicate the direction of negative class (NWZ crash). However, not all words with positive coefficients are words related to WZ, and there is no way to define words related to NWZ. For example, both LGR and SVM have the words “pushing”, “zoo”, and

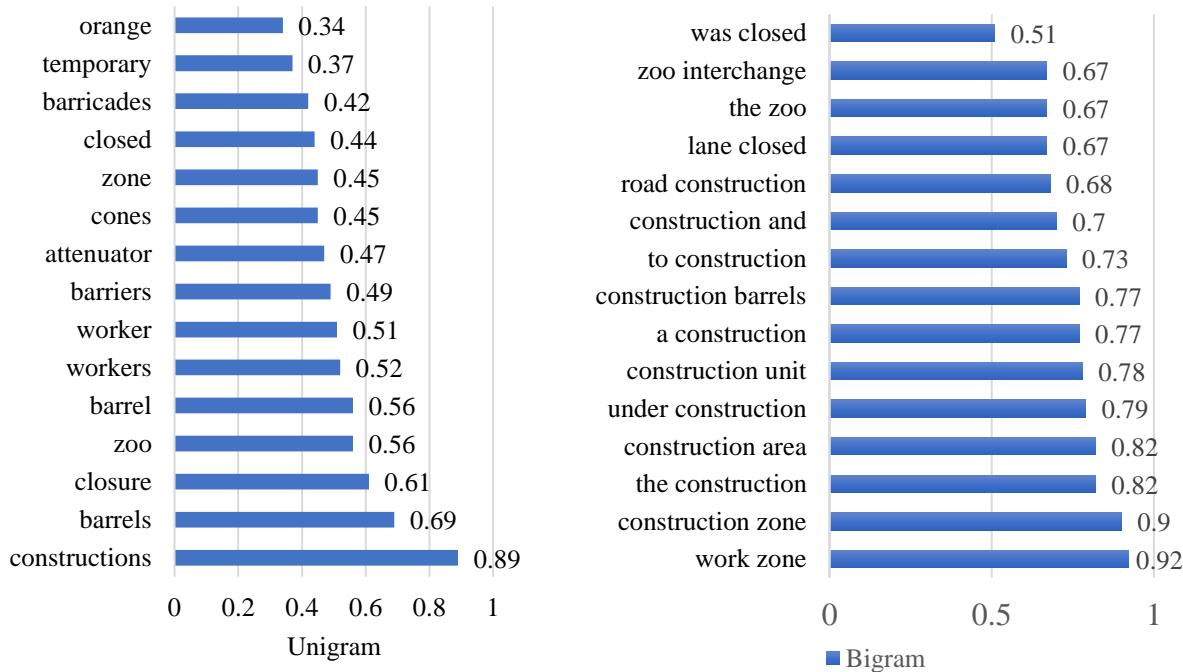
“concrete” with high positive coefficient values. The other words such as “water”, “interstate”, and “cement” in LGR list and “written”, “rain”, and “enforcement” in SVM list are present with high coefficients. It is worth noting that the core construction of the Zoo Interchange was during 2014-2018 and that is why “zoo” appears on the positive list. Through the manual review of 400 narratives of the first evaluation, it was found that many words appeared in both WZ and NWZ crash narratives, and had little to do with WZ crashes. Some words with negative coefficients such as “icy”, “snow” and “sliding” are noisy or irrelevant words. The performances of LGR in finding and weightening WZ related keywords is slightly better than SVM.



**Figure 4.6-1 Important Words Found by LGR and SVM**

For Noisy-OR, the important words were selected based on probability score (pr).

Interestingly, the words in the top 15 unigrams and bigrams of the NoisyOR word list are also common in the word lists of LGR and SVM. But, there are no tokens with negative coefficient like LGR and SVM in NoisyOR that can lead a classifier to conclude negative class. The number of positive unigrams and bigrams with probability scores greater than 0.25 are 154 and 1,665, respectively. Although LGR and SVM can identify good positive words (words with high coefficient values), the irrelevant words with negative coefficient (Figure 4.6-1) actually degrades the performance of the model.



**Figure 4.6-2 Top 15 Positive Unigrams and Bigrams Obtained by the NoisyOR**

The deep learning models (or GRU for this study) are often regarded as blackbox models because the internal mechanism they use for classification is not interpretable inability. In this study, some insights on the GRU model were gained from its output. The manual review of

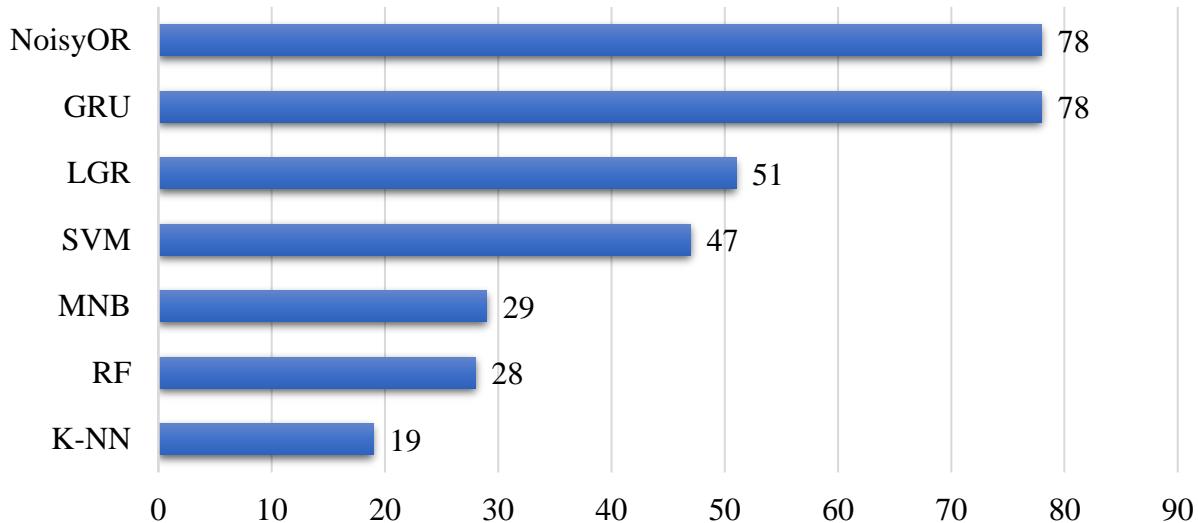
GRU's top 100 narratives helped reveal important keywords. For example, the 100 narratives with top GRU scores contained the WZ related word "construction", which means that GRU emphasized this word while classifying the narratives. The other important words in the narratives are very similar to that of LGR, SVM, and Noisy-OR.

## 4.7 Model Evaluation and Discussion

From the results obtained using the first evaluation described in previous chapter, it was found that NoisyOR achieved the highest AUC score (0.98) and GRU achieved the second highest AUC score (0.97). LGR (0.96) and SVM (0.95) provided similar AUC scores, while MNB and RF had AUC scores of 0.95 and 0.87, respectively. The K-NN achieved the lowest AUC score (0.65). It was also found that the ROC curves of NoisyOR, GRU, SVM and LGR follow similar trend. Since the differences in AUC values for these models are small, the AUC cannot be used to determine the best model. While constructing the test dataset through manual reviewing, it was found that it had many easy WZ and NWZ cases, which may be the reason for the small differences in AUC values. Another set of evaluation data may give a different outcome.

Figure 4.7-1 shows the results of the seven classifiers for the second evaluation. For each of the seven classifiers, 100 narratives ( $7 \times 100 = 700$  narratives in total) that had the highest classification scores were selected and manually evaluated them. It can be seen from Figure 4.7-1, NoisyOR and GRU perform comparably, and each detected 78 WZ crashes, the highest number of missed WZ crashes among all the methods. The performance is moderate for LGR

and SVM but is not satisfactory for MNB and RF. K-NN classifier is the worst. MNB and RF provided more than 100 cases with a classification score of 1.0, most of which were NWZ crashes. It should be noted that doing well in the second evaluation is much harder than doing well in the first evaluation because in the second evaluation the methods had to consistently not misclassify a narrative as WZ crash with a high score for a total of 82,215 narratives.



**Figure 4.7-1 Missed WZ Crash Detection Accuracy (%)**

#### 4.7.1 Analysis of Overlapping Cases for LGR, SVM, NoisyOR and GRU

It is expected that in the second evaluation, many top scored narratives will be common among the models. However, it was surprising to see that there were only 333 (about 48% of the selected 700 narratives) overlapping narratives (narratives found in top 100 of one classifier were also found in top 100 of other classifiers). The analysis of overlapping narratives provided some

insights into the classification performance of different classification methods on the same narratives. Therefore, a comparison study was conducted on the overlapping cases identified by LGR, NoisyOR, SVM and GRU; the remaining models were not included in this analysis due to their poor performances.

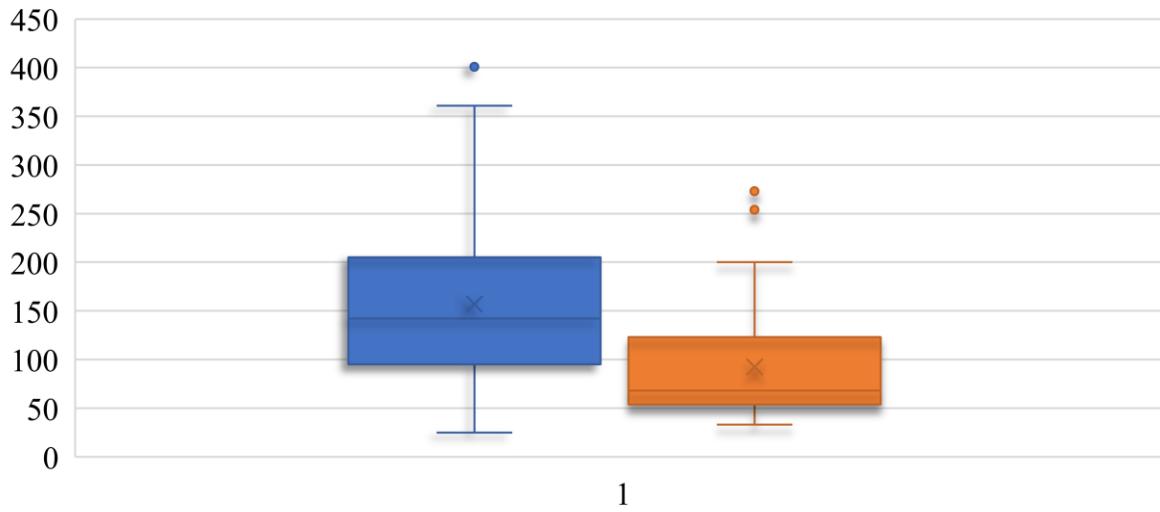
According to Table 4.7-1, there are only 43 overlapping true WZ cases between NoisyOR and GRU, which indicates that together, they found 70 different WZ cases ( $78-43=35$  for each). Through the manual review, it was found that these 70 cases contain reliable WZ keywords and are easy to classify, but when one method finds them among its top 100 narratives, the other does not. Another interesting observation is that out of the 47 true WZ cases of SVM, 45 overlap with LGR, and the model performance of SVM is very similar to LGR. There are 19 overlapping true WZ cases detected by all four classifiers (LGR, Noisy-OR, SVM and GRU) in which “construction zone”, “construction work”, and “construction lane” are the primary keywords (tokens).

**Table 4.7-1 Overlapping True WZ among LGR, NoisyOR, SVM and GRU.**

Classifier	Total	Overlapping Cases			
		LGR	NoisyOR	SVM	GRU
NoisyOR	78	45	78	41	43
GRU	78	20	43	20	78
LGR	51	51	45	45	20
SVM	49	45	41	49	20

By analyzing the 70 (i.e., 35+35) narratives as mentioned previously, it was observed that the average length of narratives of NoisyOR is longer than that of GRU. Figure 4.7-2 shows the

distribution of those narratives. The GRU uses almost all the words in the narratives whereas NoisyOR considers only positive words (i.e., unigrams and bigrams). A longer narrative may have many positive and non-positive words. As GRU considers all the positive and non-positive words, the overall classification score of the narrative may not be a large number. On the other hand, as NoisyOR only considers positive words, the classification score will be increased. For example, if a long narrative has an equal number of positive and negative words and suppose GRU regards them equally, the classification score will be 0.5 for GRU, whereas it will be more than 0.5 for Noisy-OR. In this way, a longer narrative with or without many negative words is handled better by Noisy-OR. The GRU classified smaller narratives more accurately than Noisy-OR. In the dataset, it was found that the probability score of the word “constructions” is 0.89 in Noisy-OR. If a narrative contains this positive word only with other non-positive words, the classification score of NoisyOR will be 0.89 and it will not be included in the top 100 narratives because the classification score is above 0.99 for the top 100 narratives of Noisy-OR. Therefore, the narrative is not present in the top list of Noisy-OR. On the other hand, GRU gave high weight to some of the WZ related words. Therefore, the narrative with those words has higher GRU scores and consequently they are found in the top list of GRU. In short, NoisyOR is sensitive to the number of positive words but does not consider any word as negative, on the other hand, GRU is less sensitive to the number of positive words but is affected by negative words.



**Figure 4.7-2 Distribution of Narrative Length.**

#### 4.7.2 Comparison between GRU and NoisyOR

Further investigation was done into the results of the top two performers: GRU and Noisy-OR. It was found that the top 100 cases of GRU result contained the word “construction” at least once in the narratives. It was also found that phrases in the narratives that lead to incorrect classification by GRU. These phrases can be categorized into two classes: mixed phrase and pesudo-WZ phrase. A mixed phrase refers to the combination of WZ related words and irrelevant words such as “johnson construction”, “construction at sarah's dance studio”, “construction on their new driveway”, and “construction building”. A pesudo-WZ phrase refers to the combination of WZ related words such as “construction barrels”, “construction equipment”, “construction barrier”, and “construction sign” but not in a work zone setting. Following are two example narratives that use pseudo-WZ phrases:

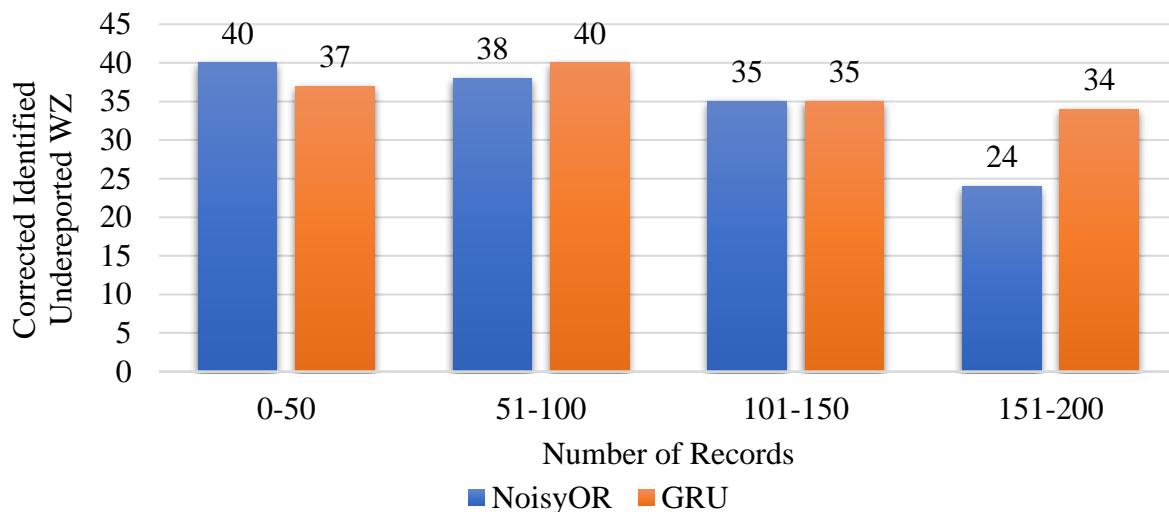
Narrative 1:“unit 1 was eastbound on i-94 in a snow storm lost control went into the median struck some **construction barrels** and ended up on the westbound side of i-94”.

In the narratives, only the presence of construction barrels does not warrant that there was a construction zone in the travel direction.

Narrative 2: “unit one was traveling westbound (north) on us 14 just south of sth 138. Unit one struck a ladder which was present in the middle of the roadway. Unit one continued to travel westbound, where he observed a white-colored pickup truck (with **construction equipment** in the bed) making a u-turn at the turn around on us 14 and netherwood st. Unit one followed the pickup truck after turning around, and later confronted the driver of the pickup truck over the ladder. The driver of the pickup truck denied the ladder was his. I was able to speak with the driver of the pickup truck a few days later, and he denied the ladder was his. He advised he works for a roofing business and owns ladders significantly larger than the one that was present in the roadway”.

In the narrative, it is clear that the construction equipment was loaded in a pickup but the accident has nothing to do with a work zone.

Among the 78 correctly classified narratives of Noisy-OR, 75 narratives contain the word “construction”. Among the 22 misclassified narratives of Noisy-OR, the word “construction” appeared in the 9 narratives, and the words “barrels”, “barrier”, and “orange” appeared several times. The pseudo-WZ phrase “construction barrels” appeared in 7 narratives. Since NoisyOR is a keyword-based classifier and does not use contextual information for classification, it fails to correctly classify these narratives. Furthermore, it was found that the words “lane”, “closed”, “attenuator”, “orange”, and “barrel” appeared several times in the remaining misclassified narratives.



**Figure 4.7-3 Accuracy of Noisy-OR and GRU**

The above analysis shows that both GRU and NoisyOR perform well, but their classification mechanisms are different. The reasons of misclassifications are very similar for some of the cases (e.g., presence of pesudo-WZ phrase). However, it is difficult to select the best classifier based on manual review of the top 100 results of GRU and Noisy-OR. That is why, the sample size was expanded from top 100 to top 200. Figure 4.7-3 shows the detection rate of missed WZ crashes in an interval of 50 data points with a maximum of 200 narratives. It was found that GRU detected 146 missed WZ crashes whereas NoisyOR detected 137 from their top 200 narratives. The detection rate of GRU fluctuates with the decrease of the classification score, but for NoisyOR, it decreases with the decrease of the classification score.

## 4.8 Summary of Observations

It was observed that there are two main challenges in identifying missed WZ crashes from narratives. The first challenge is due to the nature of WZ crash narratives. A crash narrative may be very long with several parts irrelevant to WZ, but if it mentions a word or a few words, such as “construction area”, at just one place then that would make it a WZ crash narrative. In addition, there are not many words or phrases which are indicative of WZ. This is typically not how most classes are in text classification tasks. For example, for a classification task to classify a news article as belonging to politics or not, most parts of the article will indicate that it belongs to politics and there will be plenty of words that will be indicative of that class. Many popular text classification statistical methods, such as SVM, LGR and MNB, work well with the latter types of tasks, because they tend to take into account a large number of features to make their classification decisions. On the other hand, NoisyOR can narrow down to only a few indicative words (e.g Figure 4.6-2) and only looks for their presence, and because it is a probabilistic “Or”, presence of any good indicative word is sufficient for it to classify a narrative as WZ. This is one of the reasons why the NoisyOR algorithm performed well in this study. GRU’s learning mechanism is complex and not easily interpretable, but it appears from the results that it also learned to base its decision on the presence of a few indicative words.

One more reason the nature of WZ crash narratives is different from typical text classification classes is that they can very well contain what can be in NWZ crash narratives. This is because what happens in an NWZ crash can also happen in a WZ area thus making it a WZ crash. In contrast, for example, a non-political news article will be always very different from a political news article. In other words, there are really no negative words that indicate that a

narrative is not WZ. However, methods such as LGR and SVM heavily use negative features (e.g Figure 4.6-1) which possibly confuse the methods on this task. On the other hand, NoisyOR strictly uses only positive indicators and hence is not affected. It appears from the earlier discussion related to the lengths of narratives that GRU is affected to a some extent by the negative features.

The second challenge is due to the automated created the training dataset that led to large noise as was pointed out earlier. An estimated 70% of narratives flagged as WZ do not contain anything that indicates WZ crash. This adversely affects most of the methods because they are not designed to handle so much noise. In contrast, this is unlikely to affect Noisy-OR's top unigrams and bigrams as long as there are sufficient true WZ narratives flagged as WZ. From the results, it appears that GRU was not much affected by this noise. There is also noise because many narratives flagged as NWZ are, in fact, the missed WZ crashes. Although this can potentially affect all the methods, given that a small percentage of all crashes are WZ, the extent of this noise is small. Although the above observations are specific to the task of identifying missed WZ crashes, it is likely that they will be true for the tasks of identifying other missed causes of crashes. To summarize the results, among the seven classifiers tested, MNB, RF and K-NN provide poor classification performance with the dataset. Although the AUC score, and the coefficients of WZ-related words of LGR and SVM seem promising, the performance of LGR and SVM in detecting missed WZ crashes is not satisfactory for the reasons mentioned earlier. GRU and NoisyOR are the two best performers, and their results of recovering missed WZ crashes from the reported NWZ crash narratives are comparable. Based on manual verification of the first 200 narratives of each model, GRU detected 146 WZ crashes, 9 more WZ crashes than

NoisyOR. NoisyOR can handle longer noisy narratives better than GRU. On the other hand, compared to NoisyOR, GRU can handle shorter narratives better. The word probability of a positive word in NoisyOR is prepared in such a way that if an important positive keyword is very frequent in the NWZ narratives, the word probability score is decreased. Therefore, although some short narratives of missed WZ crashes have obvious indication of WZ crashes, NoisyOR may not be able to generate higher classification scores for the narratives. On the other hand, GRU does not emphasize much on the number of occurrences of keywords in the narratives. Instead, it uses the context of the words through its sequence processing mechanism. Therefore, it is able to correctly classify short narratives. However, GRU is not able to generate high classification scores for the longer narratives. This indicates that GRU cannot handle narratives that have a few positive words with many negative words. But GRU has the advantage that it employs word embeddings which enables it to treat semantically similar words similarly in its model (for example, it will treat “barricade” and “roadblock” similarly). But NoisyOR treats every word distinctly whether they are semantically similar or not. On the other hand, NoisyOR is simple, computationally fast, and interpretable. Whereas, GRU algorithm is very complex in nature and requires fine-tuning several hyperparameters. It also requires a significant amount of time to train the model. Therefore, there is a trade-off in choosing the best model between NoisyOR and GRU.

#### 4.9 Conclusion and Recommendations

This research used NLP and ML techniques to facilitate the process of identifying missed WZ crashes from crash narratives. In order to find the best classifiers, multinomial naive bayes

(MNB), logistic regression (LGR), support vector machine (SVM), k-nearest neighbor (K-NN), random forest (RF), gated recurrent unit (GRU), and NoisyOR were tested and compared. As an experimental study, the crash narrative of the Wisconsin crash reports from 2017 to 2019 were used, where the data from 2017 to 2018 was used for training and 2019 was used for testing. For a training data set with about 70% noise (false positives) and many irrelevant words (words that do not relate to WZ), the performance of MNB, RF and K-NN were not satisfactory. Although LGR and SVM can detect many WZ-related keywords, their performance in detecting missed WZ crashes was not satisfactory. As the top two performers, GRU and NoisyOR are comparable in finding missed WZ crashes.

Further analysis suggests that two types of issues contributing to the misclassification of both GRU and NoisyOR: the mixed phrases that contain at least one highly relevant WZ word (e.g "construction building", "johnson construction ") and the pesudo-WZ phrases that contain WZ related words such as "construction barrels", "construction equipment", "construction barrier" but were used in completely irrelevant context. In other words, the narrative with pesudo-WZ phrases contains inadequate information to be classified as a work zone crash.

In addition, NoisyOR and GRU work differently in noisy narratives and short and long narratives. NoisyOR is more suitable for noisy or lengthy narratives, while GRU is suitable for shorter or less noisy narratives. In NoisyOR, an important keyword or positive word can gain less probability score (i.e. construction), which leads to generate less classification score for shorter narratives. On the other hand, GRU cannot handle longer narratives that contain many NWZ related words. GRU is complex, computationally intensive and difficult to interpret. On the

contrary, NoisyOR is very simple and theoretically sound that requires less computational power. However, to find the maximum number of missed WZ crashes, it is recommended using both the methods together.

There is still room for improvement in the future. In this work, only words were used as features for text classification. In future, a deeper analysis at the sentence level using NLP techniques could help in identifying WZ crashes more accurately. Given that the models were adversely affected by mixed phrases, in future, developing a technique to automatically identify such phrases and then removing them from the narratives is a potential avenue for future work. This work focused only on identifying missed WZ crashes, but the same techniques could be applied in future to identify other missed causes of crashes, such as distracted driving or driving under the influence of drugs. In spite of these limitations, the NoisyOR and GRU classification methods have proven their effectiveness in detecting missed WZ crashes from crash narratives. Using a combination of NoisyOR and GRU models will be a promising direction for future research.

## **Chapter 5. CLASSIFYING MOTOR VEHICLE CRASHES USING SENTENCE-BASED HIERARCHICAL ATTENTION NETWORKS**

### **5.1 Introduction**

In traffic safety analysis, researchers and safety engineers constantly rely on crash data for crash injury severity forecasting, crash risk factor analysis, crash modeling, strategic planning, and policy formulation. Therefore, it is critical to ensure that the data is accurate, reliable and complete for decision-making. Many studies have shown that the data stored in structured (also known as tabular data) field are not complete, causing one type of crashes to be underestimated while the other is overestimated (Elvik & Mysen, 1999; E Hauer & Hakkert, 1988; Sayed et al., 2021; Tsui et al., 2009; Ye & Lord, 2011a). As every crash contains unique aspects and circumstances, the officer's narrative description of crash provides crucial additional information that cannot be captured in the data fields. To many safety practitioners, crash narratives are a critical data source in addition to tables. The most common method to extract information from crash narratives is manual review. Because manual review is labor intensive and time consuming, safety practitioners are only able to review a small number of narratives to serve specific purposes. Moreover, manual review is susceptible to the experience and judgement of the reviewers. These issues necessitate a focus on the scalability and automation of crash narrative analysis.

Analyzing crash narratives falls within the scope of natural language processing (NLP) and deep neural network (DNN), both of which are the subfield of Artificial Intelligence (AI).

NLP is a computerized text data analysis technology built on a set of theories and technologies aimed at achieving human-like language processing (Liddy, 2001). The DNN is used to enhance the application of NLP by automatically learning the context of the data, thereby eliminating the need for human intervention. The characteristics of crash narrative are similar to many text data that are commonly used in NLP research in terms of grammatical structures in the sentences and temporal order in information flows; and they are usually cleaner (i.e., no special symbols or characters). However, some unique characteristics of crash narrative are challenging to develop a good model. For example,

- 1) Irrelevant information: in most of the narratives, sentences contain information of little value to safety analysis.
- 2) Missing relevant information: many crash narratives lack key information pertinent to the crash.
- 3) Highly imbalance data: due to the large number of crash types, the classification task becomes one versus all, resulting in an extremely unbalanced dataset. Deep learning algorithms have difficulty learning from highly imbalanced data (Bauder & Khoshgoftaar, 2018; Japkowicz, 2000; Krawczyk, 2016; Weiss, 2004).
- 4) Large variation in narrative length: the length of the narratives varies from a few hundred to over two thousand characters.

These issues can be effectively handled by the NLP with attention layer based RNN because the technology can detect and learn the important parts of the text through deemphasizing the less important part of the text, such as noisy words and sentences. Hence, the

attention layer is a widely-used technique for reducing the impacts of mislabeled data (Lin et al., 2016; Yin et al., 2022; Zhu et al., 2019).

The objectives of this study are twofold. First, develop a text mining method to solve the data imbalanced issue without compromising the data integrity. Second, develop an attention based deep neural network to automate crash classification. The proposed model will provide interpretable results under varying word contexts and detect important information in the text for classifying crashes. The remainder of this chapter is organized as follows. The next section discusses the crash data collection and treatment of the imbalanced dataset. Then, the proposed sentence-based hierarchical attention networks (SHAN) is introduced. Next, SHAN model results are compared with state-of-the-art DNN models using a variety of evaluation matrices and the results are discussed. Finally, the concluding section summarizes the findings and makes recommendations.

## 5.2 Crash Data Collection and Preprocessing

This research used 2019-2021 crash reports collected from the Wisconsin Department of Transportation (WisDOT) through the WisTransPortal<sup>9</sup> data hub. The proposed model was evaluated by conducting two case studies: distracted driving and inattentive driving crash classification. The reason for utilizing these two crash types is because they frequently overlap,

---

<sup>9</sup> WisTransPortal provides the computing and data management requirements of the Wisconsin Traffic Operations and Safety (TOPS) Laboratory. The scope of the project encompasses ITS data preservation, real-time traffic information services, transportation operational applications, and transportation research.

<https://transportal.cee.wisc.edu/about/>

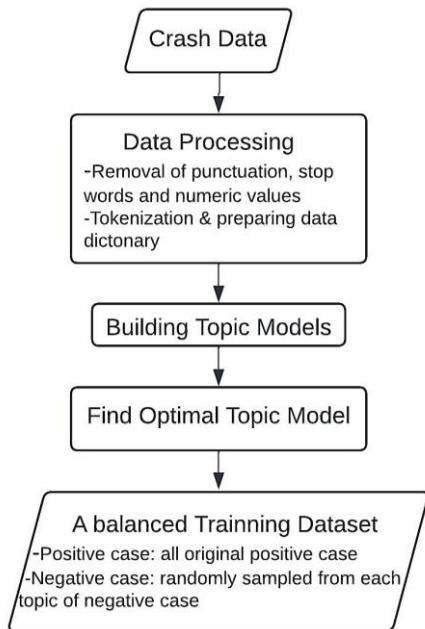
and it is challenging to distinguish between them using a model. Another reason is that their characteristics can vary depending on the type of crash, for example, one type of crash may contain more crash-related information than others.

The 3-year Wisconsin crash data contains approximately 23% involving distracted driving and 8% involving inattentive driving. The relatively low proportion of distracted driving and inattentive driving related crashes implies that both datasets are extremely unbalanced. To manage imbalance data, training data can be treated in a variety of ways, including undersampling, oversampling, and weighting minority classes, or penalizing majority classes. When the objective is to retrieve positive cases, the undersampling technique is advantageous, particularly when dealing with text classification problems. Another advantage of using the under-sampling technique is to save computer memory and require less computational power when training the model.

Down sampling a dataset needs to consider the patterns within, or a model could be highly biased. For instance, if all negative cases are taken from the topics that obtain fewest positive cases, the number of cases with common patterns will be underrepresented in the sample. If all negative cases are included from the topics that dominate positive cases, the issue of common patterns becomes acute, and the data becomes more erratic. If negative cases are considered across all topics, the patterns in the original data are preserved and the sample is more representative of the original data.

There are many hidden patterns that may not be detectable by humans in crash reports but can be found through topic modeling techniques. The Latent Dirichlet Allocation (LDA) is a widely used topic modeling technique to find hidden themes in the data and group them in a

fixed number of topics. LDA assigns topics to documents, estimates topic distributions over words given a collection of texts through discrete probability distribution, and group texts over similar themes (Campbell et al., 2003; Das, Dutta, et al., 2021; Hoffman et al., 2010). Figure 5.2-1 presents the methodological process of the data balancing techniques. Necessary data preprocessing tasks need to be completed before topic modeling, including the removal of punctuation, English stop words and numeric values from crash reports. The clean reports are tokenized into unigrams and bigrams using GENSIM<sup>10</sup> python library. The bigrams are automatically detected during tokenization and the tokens are converted into bag of words (BOW) that counts individual words or phrases.



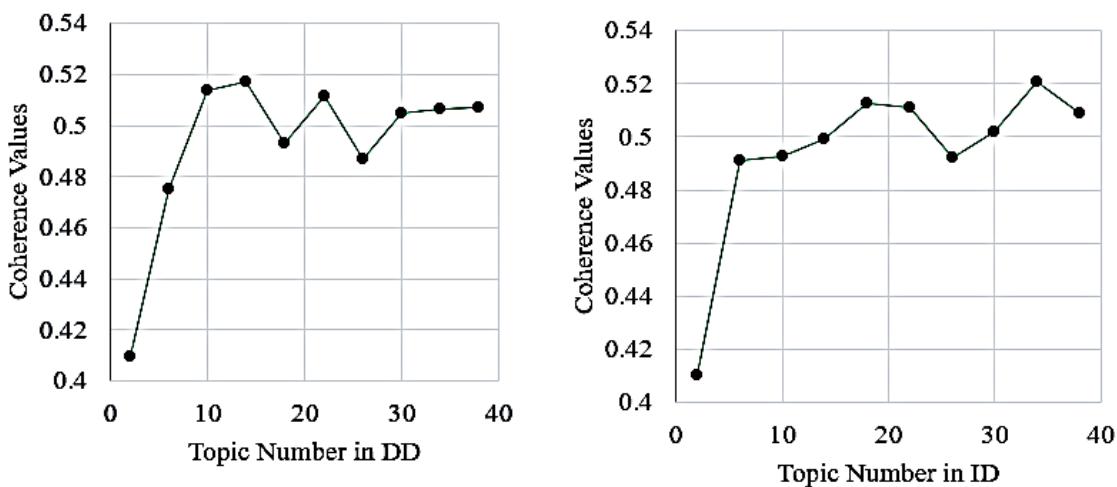
**Figure 5.2-1 Methodology for Data Balancing**

---

<sup>10</sup> GENSIM is a free python library developed for topic modeling from text data. It trains large scale semantic natural language processing models, presents text as semantic vectors and help finding semantically related documents. <https://radimrehurek.com/gensim/index.html>

The first step in topic modeling is to set an appropriate number of topics. Statistical tools, expert knowledge, and experiments can be used to determine the number of topics. In this study, the experimental approach was used to determine the optimal topic number by experimenting with a range of topic numbers and calculating the coherence value. The coherence value indicates how frequently the topic words appear in the dataset together. When multiple topics contain the same words, the topic count increases significantly. A model performs better when the coherence value is high. Therefore, the model that provided topic numbers with the highest coherence value was selected.

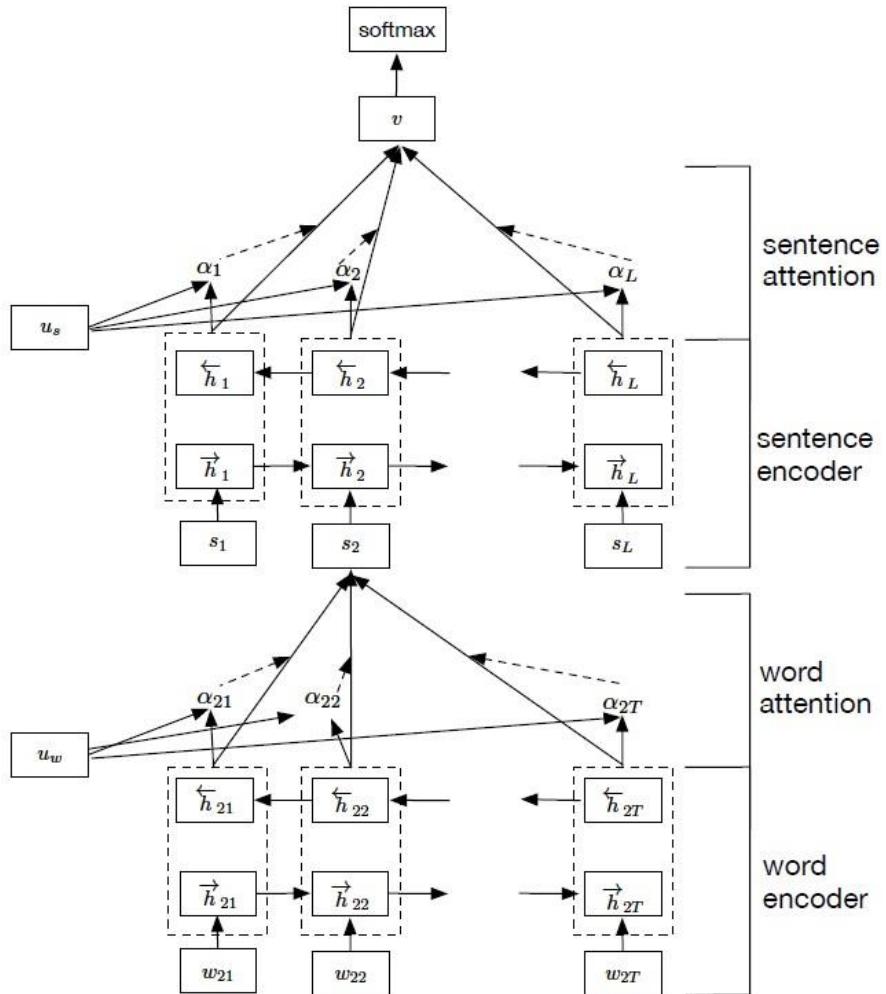
The model with the highest coherence value identified 14 distinct topics for distracted driving (DD) and 34 for inattentive driving (ID) crashes (Figure 5.2-2). It was found that both positive and negative cases of DD and ID have common topics. In other words, both positive and negative cases exhibit some similar patterns in crash reports. In this study, the balanced dataset was prepared.



**Figure 5.2-2 Optimal Topic Number Selection**

### 5.3 Sentence-based Hierarchical Attention Network (SHAN)

This study extended the most common attention model Hierarchical Attention Network (HAN) proposed by Z. Yang et al. (2016). HAN considers the hierarchical structure of sentences and words in a text and incorporates an attention mechanism capable of locating the most significant words and sentences in a text while also taking context into account (Maria Kränkel, 2019). Figure 5.3-1 shows the architecture of the hierarchical attention model.

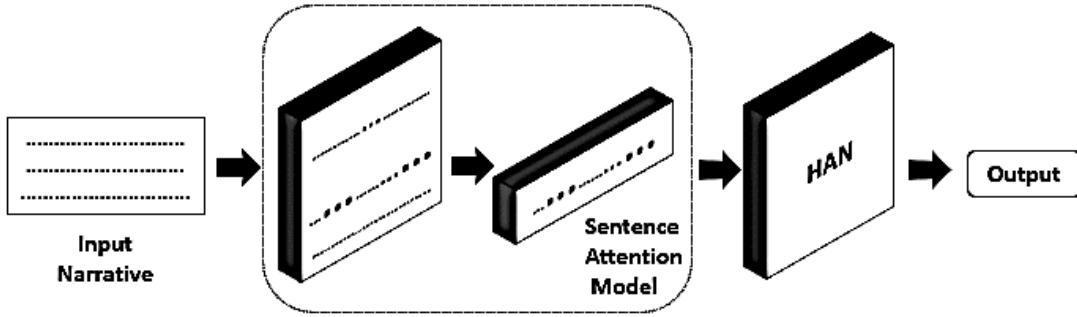


**Figure 5.3-1 Hierarchical Attention Model (HAN) (Z. Yang et al., 2016)**

The HAN model is divided into two main components: one that addresses attention at the word level and another that addresses attention at the sentence level. For word-level attention, the model takes the input in the form of embedding vector into the Bidirectional, learns the pattern of words in the sentences, and returns the word vector with updated weights to the word attention layers. For sentence-level attention, the attention layers update the weights and calculate the weighted sum of all vectors in the sentence using its own shallow neural network. This procedure is repeated for each sentence vector to create the final vector of text narrative.

As described earlier, not all the sentences in a crash report carry important information to classify the crash. A crash can be classified based on the most important sentences. Unlike news reports which contain emotions or context flows throughout the report, the crash report contains information that is not necessarily related. In other words, there could be no logical connection between two sentences. Keeping this in mind, a sentence attention model was developed and added to HAN in such a way that it only considers the most critical sentence in a crash report.

Figure 5.3-2 presents the SHAN model framework to classify a crash report. The sentence attention model takes the narrative as input, turns it into document vectors, runs the word attention layer up to sentence attention layer, and then takes the output of the sentence attention layer. The sentence attention model parses through each sentence in a crash report and extracts the most critical one, which is then sent to HAN for classification.



**Figure 5.3-2 SHAN Architecture for Classification**

This study used the implementation of HAN from (Maria Kränkel, 2019) as base of the proposed model. The following section provides model details.

### 5.3.1 Word Level Attention

In a narrative, word  $i$  in the sentence  $t$  is denoted by  $w_{it}$  and  $t \in [1, T]$ . A word embedding vector  $W_e$  provides a multidimensional vector for word  $w_{it}$ , and it is expressed as  $x_{it} = W_e w_{it}$ . The vectors of embedding layer can be initialized with some random weights or can be a pretrained word vector such as word2vec<sup>11</sup> and GloVe<sup>12</sup>. The weights of the pretrained word vector can be used as it is or can be updated during the training periods.

---

<sup>11</sup> word2vec is a family of model architectures and optimizations that can be used to learn word embeddings from large datasets. It can detect synonymous words or suggest additional words for a partial sentence. It represents each distinct word with a particular list of numbers called a vector. The vectors are chosen carefully so that they capture the semantic and syntactic qualities of words. <https://code.google.com/archive/p/word2vec/>

The output of the embedding vector  $x_{it}$  is the input of word encoder. The Gated Recurrent Network (GRU) (Bahdanau et al., 2014) was used as encoder as used by (Maria Kränkel, 2019; Zichao Yang et al., 2016). GRU has a ‘hidden state’ that transforms information to the next state. It has two gates that decide which information or words to retain or discard, resulting in relevant contexts of sentences. A bidirectional GRU reads the sentence from left to right and right to left (equation 1& 2) and then summarizes them (equation 3) to capture the context of the words from the sentences. Contexts are regarded as individual annotations of words.

$$\vec{h}_{it} = \overrightarrow{\text{GRU}}(x_{it}), t \in [1, T] \quad (1)$$

$$\hat{h}_{it} = \overleftarrow{\text{GRU}}(x_{it}), t \in [T, 1] \quad (2)$$

$$h_{it} = [\vec{h}_{it}, \hat{h}_{it}] \quad (3)$$

The word attention mechanism consists of three steps; a one-layer Multilayer Perceptron that is initialized with random weights and uses the annotation,  $h_{it}$  as input to generate an improved annotation  $u_{it}$ . The output of the  $u_{it}$  is controlled by a  $\tanh$  function, which normalizes the value between -1 and 1(equation 4). Then  $u_{it}$  is multiplied by a trainable context vector  $u_w$  (which is also known as cosine similarity) to compute attention score. The attention score is then normalized using a softmax function to obtain normalized attention weight  $\alpha_{it}$  (equation 5). Finally, the sentence vector  $s_i$  is generated using the weighted sum of the  $h_{it}$  with the attention weight  $\alpha_{it}$ , where  $\alpha_{it}$  measures the importance of word  $i$  in the sentence t (equation 6).

<sup>12</sup> GloVe is an unsupervised learning algorithm to get vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. <https://nlp.stanford.edu/projects/glove/>

$$u_{it} = \tanh (W_w h_{it} + b_w) \quad (4)$$

$$\alpha_{it} = \frac{\exp (u_{it}^T u_w)}{\sum_t \exp (u_{it}^T u_w)} \quad (5)$$

$$s_i = \sum_t \alpha_{it} h_{it} \quad (6)$$

### 5.3.2 Sentence Level Attention

The output of the word attention  $s_i$  is the input of sentence encoder, which disregards the necessity of having an embedding layer at sentence level (Maria Kränkel, 2019; Zichao Yang et al., 2016). The sentences are also encoded by bidirectional GRU (equation 7 & 8) which are then concatenated to obtain annotation of each sentence. Then,  $h_i$  summarizes the neighboring sentences of sentence  $i$  and capture the context of the sentence  $i$  (equation 9).

$$\vec{h}_i = \overrightarrow{\text{GRU}}(s_i), i \in [1, L] \quad (7)$$

$$\overleftarrow{h}_i = \overleftarrow{\text{LSTM}}(s_i), i \in [L, 1] \quad (8)$$

$$h_i = [\vec{h}_{it}, \overleftarrow{h}_{it}] \quad (9)$$

The sentence attention mechanism identifies the most influential sentences in the narratives. The annotation  $h_i$  is multiplied by a trainable weight vector  $W_s$  to get a sentence level context vector  $u_i$ . The weight of the vector  $W_s$  and  $b_s$  are randomly initialized and adjusted during training period (equation 10). Then the context vector  $u_i$  is multiplied by a trainable context vector,  $u_s$  and normalized again using a softmax function to obtain the weight vector  $\alpha_{it}$  (equation 11). The

weight vector  $\alpha_i$  measures the importance of sentence  $i$  in crash report. Finally, the vector of crash report  $c_r$  is generated using the weighted sum of the  $h_i$  based on  $\alpha_i$ .  $c_r$  summarizes all the important sentences in a crash report.

$$u_i = \tanh (W_s h_i + b_s) \quad (10)$$

$$\alpha_i = \frac{\exp (u_i^T u_s)}{\sum_i \exp (u_i^T u_s)} \quad (11)$$

$$c_r = \sum_i \alpha_i h_i \quad (12)$$

### 5.3.3 Classification

After generating the vector of crash report, the model uses the following objective function (equation 13) for narrative classification. The weight vector  $W_c$  and bias  $b_c$  are also trained during the training period.

$$\hat{y} = \text{softmax} (W_c c_r + b_c) \quad (13)$$

where  $\hat{y} = [\hat{y}_0, \hat{y}_1]$  is the predicted probability vector with  $\hat{y}_0$  and  $\hat{y}_1$  indicate the predicted probability of label being 0 (positive case) and 1 (negative case) respectively. In order to minimize loss as well as optimize the model, the following categorical cross entropy function (equation 14) was used where  $y$  represent ground truth and  $\hat{y}$  is the prediction. The  $i$  denotes the label of crash report  $y$  and  $c \in [0,1]$ .

$$L(\theta) = - \sum_{i=1}^c y_i \cdot \log \hat{y}_i \quad (14)$$

where  $\theta$  is the model parameters of the network that are learned through an activation function.

## 5.4 Data Preprocessing and Model Parameter Setup

The data was cleaned before feeding it in the classification models. All the punctuation and English stop words were removed from the narratives. Then, the lemmatization was applied to determine the lemma of a word based on its intended meaning in the sentence. The majority of sentences with a length of less than three are road names. Thus, they were removed from the data. The average number of words in the sentences (also known as sentence length) was 20 and the average number of sentences in the crash reports was 6. Considering that, the length of input vector was set to 20 and 10 for word and sentence level attention layers, respectively. The pretrained weights of the Glove model were used for the words. The dimension for word embedding was set to 100, while the dimension for BI-GRU was set to 50.

Following the GRU layer, a 100-dimensional dense layer was used and a ReLU (rectified linear unit) activation function was linked to the attention layers. ReLU enables faster and more effective training of deep neural networks on large and complex datasets than sigmoid or similar activation functions. Following the sentence level attention layer, a drop-out rate of 0.5 was applied with a fully connected dense layer, which resulted in classification scores for both positive and negative cases. Additionally, 64 mini-batch sizes and 7 epochs were used to train the model.

In a police crash report, each crash record is manually labeled to a specific crash type by a police officer and some crashes are not properly recorded. The goal of classifying crash reports is to recover missing crashes through a model that maximizes the true positive rate (Recall) while minimizing the false negative (FP) and false positive (FN) rates. In addition to SHAN, HAN, GRU, and several statistical and ML models were developed for comparison. The statistical model includes NoisyOR from (Sayed et al., 2021) and the ML includes Naïve bias, Support vector machine, Random Forest, XGBoost, Logistic regression and K-nearest Classifier that are commonly used for text classification. The optimal values for F1, Recall, false positive rate (FP), and false negative rate (FN) for each model were determined through multiple experiments.

The results of top three best performance are presented in Table 5.4-1. It shows that proposed SHAN model outperforms GRU and HAN in both ID and DD crash classification. Although HAN has a smaller false positive rate of 0.06 for DD, the improvement is marginal.

**Table 5.4-1 Model Evaluation for Inattentive and Distracted Crash Classification**

	Models	F1	Recall	Accuracy	FP rate	FN Rate
Inattentive	GRU (0.65)	0.51	0.80	0.73	0.29	0.19
	HAN (0.60)	0.47	0.74	0.78	0.33	0.22
	SHAN (0.80)	0.66	0.89	0.85	0.16	0.11
Distracted	GRU (0.55)	0.42	0.67	0.66	0.34	0.33
	HAN (0.85)	0.73	0.74	0.90	0.06	0.26
	SHAN (0.85)	0.74	0.78	0.90	0.07	0.22

Note: the value in parenthesis with model names indicates cutoff values for classification scores.

GRU performed worst in all aspects because it took the entire text as input, which included a lot of irrelevant information. In contrast, the attention mechanism assigned more weight to relevant parts of the text than to less relevant parts, thereby improved model

performance. Since there were many false positives in the training dataset and not all the words (i.e., road name) in a text were relevant to crash, the SHAN outperformed GRU. SHAN also performed better than HAN because when classifying a text, SHAN used the sentence with the highest weight, rather than every sentence in the text.

The experimental results of two distinct types of crashes (DD and ID) demonstrate that SHAN is effective at identifying crashes in crash narratives. As the performance of SHAN is consistent across two different crash types, SHAN is more stable than other models. This also implies that the treatment of data imbalance was as effective as expected.

## 5.5 Result and Discussion

A text document contains multiple sentences, and each sentence consists of many words. However, in a text document such as in a crash report, all the sentences and words do not carry same weight. For example, in the following police crash report, only the bolded sentence is the most important because it contains the most important words to classify the crash as a distracted driving (DD) crash.

“Unit 2 was stopped at the red traffic signal on w. Paradise dr. At s. Silverbrook dr. In the left turn lane to travel northbound on s. Silverbrook dr. Unit 1 was traveling eastbound on w. Paradise dr. And had merged into the left turn lane to travel northbound on s. Silverbrook dr. **Unit 1 had begun to slow down and then dropped something, so the operator looked down and reached for it.** Unit 1 misjudged the distance to stop before unit 2 and then struck the rear bumper of unit 2 with its front bumper.”

The attention mechanism of SHAN was effective in extracting such important sentences from the text based on the presence of important words such as “dropped something”, “looked down”, and

"reached for". It was also successful at assigning different weights to the same words when the words were used in different contexts. For instance, the word "looking" has been used in two distinct contexts in the following two crash reports:

"Vehicle 1 drove off the roadway due to **looking** at a cell phone, drove off the road, hit a guardrail, overturned and rolled down a ditch and ran into trees."

"Unit 1 was n/b on sthy 57 **looking** to make a right turn to go east on cty rd a. The roadway was covered in snow, which caused the vehicle to slide straight rather than complete the turn. The vehicle struck the guardrail face with its front end."

The first case is classified as a DD crash, while the second case as a non-DD crash. In the first example, the word "looking" was given more weight (importance) than in the second example.

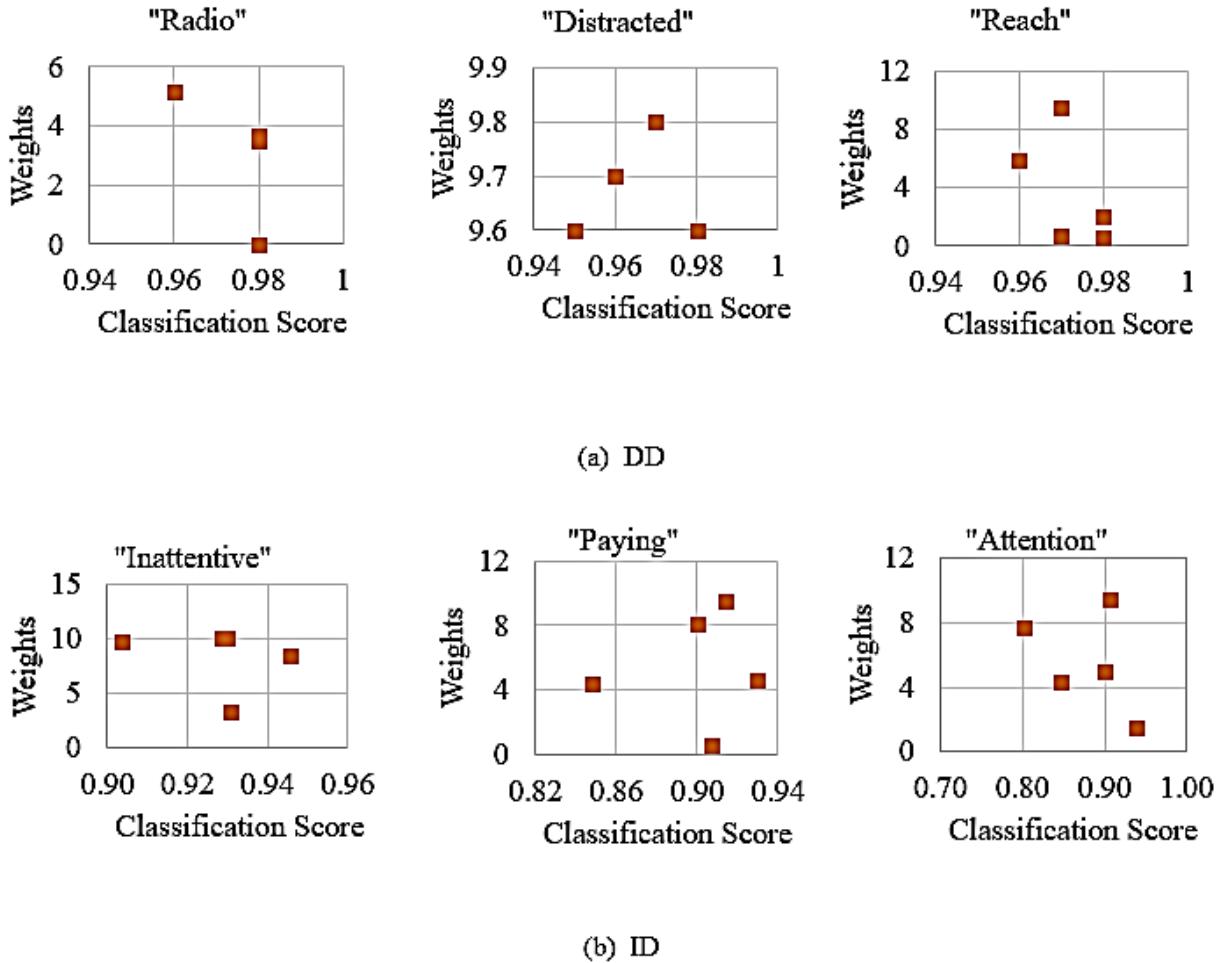
Visualizing the results with the data given in the crash report can simplify the crash report investigation and better explain the results. For example, highlighting the most weighted sentence and its words in a narrative can substantially speed up the process of reviewing and identifying crash contributors. In Table 5.5-1, the correctly classified crash reports with most weighted sentences and words are highlighted for ID and DD crashes, respectively. The result clearly demonstrates that the proposed SHAN model identified and utilized important sentences and words in the crash reports to identify DD and ID crashes. It also indicates that the proposed model successfully classifies crash reports regardless of the length of the crash report and the position of high weightage sentences within the report. Additionally, the weighted words can form a phrase or can be found individually in the report, suggesting that the model utilized all unigrams, bigrams, trigrams, etc.

**Table 5.5-1 Example of Most Weighted Sentence and Words Visualization for Distracted (DD) and Inattentive Driving (ID) Crashes**

DD	<p>~on this date and time, i ofc rasmussen of the waupun pd, was dispatched to an accident involving 4 vehs, that occurred near the waupun high school, on e lincoln st in the city of waupun, dodge county. all four veh's were traveling westbound on e lincoln st. the individuals involved all stated that traffic was backed up on e lincoln st near grove st due to the school getting out. unit 3 and unit 2 were stopped awaiting traffic to proceed, when unit 1 collided with the rear end of unit 2, causing unit 2 to collide with the rear end of unit 3. unit 1's driver stated that he was <b>distracted</b> while <b>watching</b> some <b>pedestrian</b> on the sidewalk to the north of the roadway , which caused him to not be able to stop his veh before crashing into the rear of unit 2 after this collision unit 4 came from the east , traveling westbound and collided with the rear of unit 1. this collision caused unit 1 to further collide with unit 2, causing unit 2 to further collide with unit 3. unit 4's driver stated she was distracted by the pedestrians on the sidewalk to the north, which caused her to not be able to stop in time before colliding with unit 1. unit 1 and unit 4 were towed due to damage. unit 2 driver reported having neck pain from the collision. unit 1 and unit 4 driver's were cited for inattentive driving.~~</p> <p>~unit 1 was traveling north. <b>operator states he reached down</b> to pick something up and drifted off the shoulder .. the snow bank pulled the bus off into the ditch. operator attempted to correct but ended up sliding sideways into a utility pole that broke off and landed across the top of the bus causing damage to the roof.~~</p> <p>~driver of vehicle 2 stated he was stopped in the inside lane of southbound traffic in the 2000 block s central ave when vehicle 1 crashed into the back of his vehicle. <b>driver of vehicle 1 stated she was looking</b> down for a moment to flick an ash from her <b>cigarette</b> and when she <b>looked back</b> up she noticed vehicle 2 stopped .. she stated she was unable to stop in time to avoid a collision with vehicle 2. driver of vehicle 1 also said her brakes were bad.~~</p> <p>~unit 2 was traveling east bound on w wisconsin st and was slowing to a stop for traffic. <b>unit 1 was traveling east bound on w wisconsin st</b> driver of unit 1 stated that he was <b>talking</b> to the passenger in his vehicle and did not notice unit 2 slowing to a stop .. unit 1 struck unit 2. unit 2 then pulled his vehicle into a safe area prior to my arrival. officer d pomeroy #7~~</p>
	<p>~i responded to a two vehicles crash that occurred on i-41 northbound at hampton avenue. upon arrival unit 1 was in the right distress lane and unit 2 was in the ditch on the right shoulder of the hampton on ramp. the passenger of unit 2 reported pain all over her body and was transported by ems ground to froedert hospital . both the operators of unit 1 and unit 2 did not report any injuries. the operator of unit 1 was identified with a wisconsin driver's license. he stated that he was traveling in the far right lane at about 60 mph. <b>he looked down at his cell phone for gps direction</b> and when he looked back up , a vehicle was in front of him .. he stated that it looked like the vehicle was stopped in traffic and he could not stop in time and hit that vehicle. the operator of unit 2 was identified with a wisconsin driver's license. he stated that he was traveling in the right lane underneath the bridge and was traveling at the posted speed limit. he then observed a vehicle from his rear view mirror, he further stated he believed the vehicle in his rear view mirror was going over 100 mph when it struck him from the rear. upon further investigation, both vehicles were traveling in lane 3 northbound of i-41 approximately at hampton avenue. unit 1 was distracted and failed to realized that there was a vehicle in front of him. unit 1 was unable to slow down and rear ended unit 2, forcing unit 2 to lose control and struck both a light pole and a metering light. unit 2 had major disabling damage to the entire vehicle. unit 1 had major disabling damage to the front end. both vehicle were removed by n&amp;s towing. 22 photos taken.~~x</p>
	<p>~on the above date and time, unit 1 was operating eb on e mason. unit 1 driver admitted to not paying attention and crashed into unit 2, which in turn crashed into unit 3. two juvenile passengers in unit 1, and the driver of unit 2 were transported to nearby hospitals. unit 3 had little to no damage and was able to drive away. <b>driver 1 was cited for inattentive driving.</b></p>
	<p>~unit 1 was traveling westbound on ridge st, when unit 1 crashed into the back of unit 2. unit 2 was stopped in the lane of travel. update to original crash report. unit 2 contributing factors changed to, other contributing factors, and unit 2 driver distractions changed to unknown if distracted. unit 2 operator alcohol test was changed to test given, submitted to a pbt test with results of 0.00. several days after the accident unit 2 operator informed me that he had a back injury, however the injury box will remain unchanged because at the time of the accident, unit 2 did not report any injuries. <b>unit 1 operator said he was looking</b> into the fields when he <b>looked back forward</b> and <b>seen unit 2 stopped</b> in the road. unit 1 said he could not slow down to avoid hitting unit 2. unit 2 said that his turn for the on ramp of ush 151 caught up to him too fast, leading unit 2 to an abrupt stop in the lane of travel. unit 2 said while deciding what to do, he seen headlights in his mirror and tried to get out of the way but was unsuccessful.~update~</p>
	<p>~via dispatch, accpdo on i-94 westbound at 16th st on january 25, 2018 at 1553 hours upon arrival: unit 2 was stopped in traffic in lane 5, and unit 1 was pulled over into the grass. the operator of unit 1 stated that she was following the vehicle ahead of her in heavy traffic. she stated that she looked behind her to merge to her right, and that she then hit the car in front of her. she advised that she did not realize the car in front of her had stopped. she reported no injuries. the operator of unit 2 stated that she was in the far right lane, in heavy stop and go traffic. she said that she had just come to a stop due to traffic ahead of her, when she was hit from behind. she reported no injuries. upon investigation: unit 1 was following unit 2 in lane 5 in heavy stop and go rush hour traffic. <b>unit 1 wanted to merge to the left pass unit 2</b> , and therefore she took eyes from the road way in an <b>inattentive manner</b> to check for traffic before she merged to lane 4 .. unit 2 was forced to stop due to traffic stopping ahead of her, and at this time unit 1 was not looking at the road in front of her. unit 1 then rear ended unit 2. no assisting squads no injuries no tows no gov't property struck no hazmat~~x2, 25</p>
ID	<p>~unit 1 was completely stopped in lane 1 on n. bluemound dr, south of w. college ave. unit 1 was waiting to make a left hand turn onto college. unit 2 was behind unit 1. <b>unit 2 was behind unit 1. unit 2 stated he had been daydreaming</b> and thought unit 1 had completed their turn. unit 2 rear ended unit 1.~~</p>
	<p>~unit 1 and unit 2 had been traveling on university avenue in the right eb lane; unit 1 was behind unit 2. as they approached the driveway to moka coffee at 5227 university avenue, there was a vehicle in the roadway stopped and turning right into the driveway. unit 2 stopped behind this vehicle. the <b>driver of unit 1 admitted she glanced</b> away from the roadway .. when she looked back, unit 2 was stopped in front of her. unit 1 then ran in the back of unit 2.~~</p>

Note: orange color=most important sentence, red color= important words, green color=word is not present in the trained SHAN.

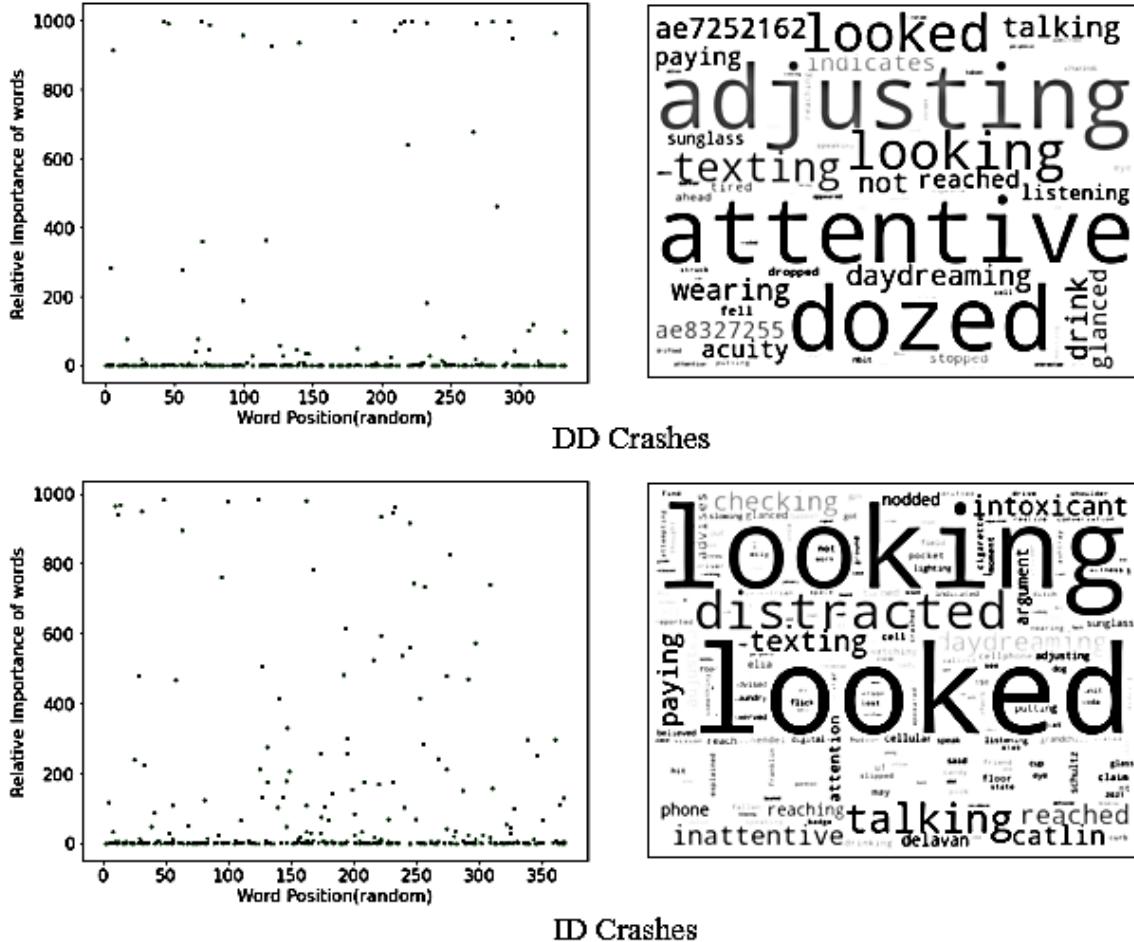
Figure 5.5-1 shows attention weights of the three important words such as “Reach”, “Distracted” and “Radio”. Similar investigation was also conducted for inattentive crashes. As illustrated in the graphs, the weights of each word were changed when they were used in different contexts, as captured by SHAN.



**Figure 5.5-1 Attention Weights of Words in Various Contexts**

Identifying DD and ID is difficult because same words could appear in both cases (e.g., “looking” or “paying”). The proposed SHAN model is able to differentiate the common words by applying different weights, as illustrated in word clouds in Figure 5.2-2. As seen in the figure,

the model captured crash-related words from the narratives and assigned various weights to them for DD and ID classifications. Also, it shows that only a limited number of words in the upper range (e.g., words with relative importance >400) are in the vocabulary to classify DD and ID reports.



**Figure 5.5-2 Words with High Attention Scores and Their Relative Importance**

## 5.6 Conclusion and Future Works

Crash narrative contains rich and valuable information to improve traffic safety. In the meantime, the data is extremely unbalanced and noisy in terms of incorrect labeling, irrelevant words and sentences. The attention layer-based deep learning model provides an opportunity to deal with noisy text data, incorrect labeling, and identify the important words and sentences. However, this type of model has never been applied for text analysis in the traffic safety field. To fill this gap, this research developed a novel Sentence-based Hierarchical Attention Network (SHAN) that can properly classify motor vehicle crashes by different types from crash narratives. SHAN was developed using Bidirectional Gated Recurrent Units (BI-GRUs) as word and sentence encoder, and two word and sentence level Attention lawyers. SHAN's performance was compared with GRU and HAN, and other state-of-the-art ML models. The SHAN model outperformed them in identifying distracted and inattentive crashes from crash narratives.

Data imbalance is a common challenge for classification problem where the number of negative cases is overwhelmingly greater than the positive ones in this study. This issue was handled by undersampling negative cases. To preserve the data integrity, the LDA method was applied to group narratives into distinct topics based on their similarity and negative cases were randomly sampled from each topic. The results showed that the SHAN was effective and stable in classifying crashes using the sample data. Additionally, the visualization of the SHAN outcomes demonstrated its effectiveness in identifying key sentences and words in narratives, which could accelerate the crash review process.

Recently, some advanced pretrained models such as XLNet (Zhilin Yang et al., 2019), ERNIE(S. Wang et al., 2021), Transformer (Vaswani et al., 2017), and BERT(Kenton et al.,

1953) have showed promising result in text classification in other domains. Leveraging SHAN model with them will be the future research direction.

## **Chapter 6. DESIGN AND DEVELOPMENT OF TEXT INTELLIGENCE**

### **SOLUTION TO IMPROVE HIGHWAY SAFETY.**

#### **6.1 Introduction**

Despite significant advances in vehicle technologies, safety data collection and analysis, engineering improvements, over 30,000 people die in traffic accidents in the United States of America each year (NHTSA, 2020). What is alarming is that the trend of fatal and serious injury crashes seems to move in the wrong direction. A 12-year statistic from 2011 to 2021 shows that the fatality rate increased each year over the decade; particularly, in 2021 the actual rate of fatalities exceeds the projected rate. The challenge of vision zero to end traffic-related fatalities and serious injuries is enormous as crashes are complicated: they are not only associated with a multitude of factors but also composed by a chain of actions and events. Although the common safety practice is to identify and remove risk factors, breaking the chain of events could be another way to prevent a crash from reaching its most harmful event. The patterns of types and sequences of events taking place before the most harmful event present opportunities for safety intervention, but it is difficult to identify them from millions of crash reports filed every year.

In Model Minimum Uniform Crash Criteria (MMUCC), a guideline that presents a model minimum set of uniform variables or data elements for describing a motor vehicle traffic crash by the National Highway Traffic Safety Administration (NHTSA), crash events are recorded as “harmful event” (e.g., “C7. first harmful”, “V21. most harmful”) and “V20. Sequence of Events” (MMUCC, 2012). First harmful event is defined as “first injury- or damage-producing events of

the crash” and most harmful event is defined as “the event that resulted in the most severe injury or if no injury, the greatest property damage involving a motor vehicle”. Both are further classified as non-collision harmful events (e.g., fire/explosion, immersion, jackknife, overturn/rollover), collision with person, motor vehicle, or non-fixed object, and collision with fixed object through a list of 41 attribute values. “Sequence of Events” are events “in sequence related to a motor vehicle, including non-harmful events, non-collision harmful events, and collision events”. In addition to all 41 attribute values for non-harmful events, there are 10 non-harmful events such as cross centerline, cross median, end departure (T-intersection, dead-end, etc.), downhill runaway, equipment failure (blown tire, brake failure, etc.), ran off roadway left, ran off roadway right, reentering roadway, separation of units, and others. The rationale is to present complete information about the crash during the crash evaluation.

Why is it important to refer to crash narratives to understand crash sequence while such information seems available in structured data? There are several issues in structured data that undermine the value of crash sequence, such as:

- 1) The current MMUCC design is imperfect. The structured data fields cannot capture all the information, some of which is important and useful.
- 2) The available data fields may not be filled out properly.
- 3) The analyst may not have the ability to query all the relevant information pertaining to a crash due to the complicated design of MMUCC.

Fortunately, such information may be stored in crash narratives. Considering the issues with structured data fields, extracting crucial information pertaining to crash events from a narrative

can help us better connect crash contributing factors to safety countermeasures. A few studies have been conducted on crash sequences using structure data and crash narratives (Gao & Wu, 2013; Kwayu et al., 2021; Song et al., 2021; K. F. Wu et al., 2016, 2018). Although these studies repeatedly discovered valuable information in crash sequences for developing safety measures, significant amount of effort to manually extract information from narratives hinders the progress in this area. The goal of this research is to develop text intelligence solutions using NLP technologies to extract information from crash narratives and generate deep insights for highway safety applications. The algorithm has been developed to subsidize manual effort of generating crash sequence from crash narratives and find missing events that are not captured in structure data.

## 6.2 Crash Narrative Analysis: Issues and Challenges

The DT 4000 crash reports for the year 2020 were collected from the Wisconsin department of transportation (WisDOT<sup>13</sup>). For simplicity, this study used injury and fatal related single-vehicle crashes. DT4000 records 58 different crash events in the sequence of events fields (SEQEVT[1,2][A,B,C,D] ). The numbers in the first pair of parentheses represent the UNIT number, where the UNIT is any vehicle, bicycle, pedestrian, or piece of equipment involved in an accident. The letters in the second parenthesis represent the first four harmful events in order. For example, the event sequence of UNIT 1 will be presented as SEQEVT1A, SEQEVT1B, SEQEVT1C, and SEQEVT1D. For the rest of the sections, crash events are presented in italic.

---

<sup>13</sup> The Wisconsin Department of Transportation is the U.S. state agency responsible for planning, constructing, and maintaining the state's highways. <https://wisconsindot.gov/Pages/home.aspx>

The task of generating crash sequences is difficult due to the complex nature of crashes, complicated structure of crash narratives, and generalized form in event definition. The following are some of the key findings in crash narratives:

- 1) An event can consist of single words or a group of words that includes direct objects such as *light pole*, *traffic sign*, or action words such as *cross centerline*, *run off roadway*, *left turn* and *right turn*. An event cannot be extracted based only on verbs or objects.
- 2) Events are not explicitly present in the narrative such as *run off roadway* in the following:

“Unit 1 was nb on pine hill rd s of saxeville rd. Unit 1 entered the east ditch and had the passenger side rear corner collided with the utility pole. Unit 1 continued on for about 1 mile. Driver of unit 1 realized he had damage and reported the incident. The utility pole had a minor scuff mark on the wood and did not appear to have any structural damage.”

This narrative did not explicitly mention that the vehicle drove off to the right side of the road. The information can be extracted only from the driving direction and the location of the ditch. The same problem was found with *cross centerline* events.

- 3) Missing information from structured data fields or narrative fields. For example, *ditch* was recorded in the structure data but not in the narrative. Therefore, the event *ditch* cannot be extracted from the narratives.

“Unit 1 was south on 76th st/cth u approaching 7 mile rd. Unit 1 began to enter gravel shoulder of s/b lane. Driver over corrected, causing vehicle to slide east, across 76th st into a field. During the slide, the vehicle turned onto its driver's side.”

- a. Inconsistent unit description. As can be seen from the three examples above, a unit is represented by different words in narrative like 'driver', 'vehicle', 'unit 1', 'she', 'operator'.

All the above-mentioned issues made algorithm difficult to extract relevant information from the narrative to construct the events of crash sequences.

### **6.3 Algorithm**

Automatically extracting sequence of events consisting of words and phrases is a complex and daunting task. No previous research in transportation or other domains has accomplished this. In the transportation field, (Gao, 2022; Song et al., 2021) attempted to generate sequence of events from crash narratives. The sequences of events in (Song et al., 2021) were prepared by manually reviewing a few hundred crash narratives of autonomous vehicles. The other study conducted by Gao (2022) used NLP techniques to extract verb-based action words. The former study is not applicable for large datasets. The later study shows some efforts but did not go any further than action verbs. In this study, crash sequence algorithms were developed using several NLP and machine learning (ML) based techniques.

First, Part-of-Speech tagging (PT) algorithm was developed using NLP techniques. The purpose is to investigate the part-of-speech tags of the event-related keywords and their supporting words. Consequently, a list of action words and their outcomes were created using

python Spacy<sup>14</sup> package. The words in the narratives having those POS tags were extracted to construct the sequences of events. However, this algorithm suffers from extracting unimportant words while missing important ones.

Next, two advanced algorithms pattern matching with part-of-speech tagging (PMPT) and dependency parser (DP) were developed. The details of the PMPT and DP are presented in Appendix. As the training set for PMPT, 100 narratives were randomly selected, and crash sequences were generated based on the training narratives. The goal is to find a pattern of POS (Part-of-Speech) tags in the narrative that could be used to find the events. Based on the crash sequences, a list of POS tags was automatically generated, which essentially included the POS of words and phrases (see Appendix). This algorithm provided improved results (than PT) but missed many events if they did not follow the pattern of the tags. While PMPT only looks for matching pattern of POS tags in the narrative, DP searches the relationship between action words and the outcomes using dependency parser trees. Therefore, DP is better at finding action words and their outcomes than PMPT. However, DP failed to construct sequence of crash events that consist of different forms of words because events can be a single word (e.g., curb), a pair of words (e.g., left turn) and a pair of action verbs and words (e.g., cross centerline). Therefore, the hybrid generalized (HGEN) algorithm was developed. The HGEN works in a four-step process: create event-related keywords, create action keywords, identify contents by unit numbers and create sequence of event by unit numbers.

---

<sup>14</sup> Spacy is an open-source, free Python library for advanced Natural Language Processing (NLP). Spacy is intended for production use and facilitates the development of applications that process and "understand" large volumes of text. It can be used to construct information extraction and natural language understanding systems, as well as to prepare text for deep learning. <https://spacy.io/usage/spacy-101>.

### 6.3.1 Step 1 Create Event-related Keywords

To extract event information from narratives, a logical OR operation was performed on the fields of event sequence in the structure data. For a crash, if a crash event field contains an event E, the narrative of that crash is labeled as E; otherwise, NE. Then, the word probability equation (equation 1) from (Sayyed et al., 2021) is applied to calculate the probability of an event given a word. Thus, the problem of extracting event information can be considered as a problem of binary classification. The words in positive narratives are candidates for the event E if they occur more frequently in positive narratives than in negative narratives.

$$\text{Probability of a word } (W_E) = \frac{\text{Positive Count}(C_{wE})+1}{\text{Positive Count}(C_{wE})+\text{Negative Count}(C_{wNE})+2} \quad (1)$$

where  $C_{wE}$  represents the frequency of word  $W$  appears in positive narratives (E). Likewise, the Negative Count  $W_{NE}$  represents the number of occurrences of  $W$  in negative narratives (NE). A simple version of Laplace smoothing was applied, which assumes  $W$  occurs at least once in a positive narrative and a negative narrative to control unrealistic probability score (1) in case there are only few positive cases and no negative cases. As part of data processing, all the words were lemmatized and any positive words that occurred less than ten times in the positive narratives were eliminated. After generating positive words for an event, the most relevant unigrams and bigrams to the event were selected.

The algorithm first searched bigram then unigram to avoid repetition of the occurrence of an event. The list of candidate words and phrases extracted from narratives can be found in Appendix. While extracting events from narratives, events such as *motor vehicle in transport*, *other roadway*, *other fixed and non-fixed object*, *other post*, *other non -collision*, *fell/jumped*

*from motor vehicle, thrown or falling object, other non -motorist, struck by falling or shifting cargo or anything set in motion by motor vehicle, other traffic barrier, downhill runaway, equipment failure, separation of units and unknown* were ignored because no specific information or pattern was observed in their respective unigram and bigram lists. In addition, all animal related events such as *domesticated and non-domesticated animal* were converted into a single type of event *animal* and the *left and right run off roadway* events to *run off roadway*.

### **6.3.2 Step 2 Create Action Keywords**

In a narrative, a single word or phrase may serve multiple purposes. Therefore, event information cannot be extracted from narrative using only words or phrases. It was observed that each event is accompanied by one or more action verbs, with the exception of action verbs that represent events themselves, such as "turn over," "turn left," and "cross centerline". All action words associated with each event were extracted from the corresponding positive narratives.

### **6.3.3 Step 3 Identify Contents by Unit Numbers**

A narrative may contain one or multiple units. Therefore, it is necessary to identify which information in a sentence of narrative belongs to which unit. In this step, Spacy, a python package, was used to create natural language processing (NLP) pipeline. Spacy has shown overall best performance in real world application and provides many pre-trained tools which are

easily customizable. The “en core web trf”<sup>15</sup> pipeline, which is a pre-trained English transformer consisting of a tagger, a parser, an attribute ruler, a lemmatizer, and a name entity recognition, was applied. In order to determine the unit number, the words describing the action, and the events themselves, their taggers, parsers, and lemmatizers were used.

#### 6.3.4 Step 4 Create Sequence of Event by Unit Numbers

As presented in Figure 6.3-1, Algorithm 1 extracts event sequences from a narrative by using the dictionary of event-related words prepared in Step 1 and the dictionary of action-related words prepared in Step 2. The algorithm extracts event sequences from crash narratives by separating the content in a narrative by unit number. To do that, the narratives are converted into Spacy tokens that include words with all the attributes mentioned in Step 2. After that, the algorithm first determines if the word is related to an event, and then determines if it is also associated with an action. If a match is found, data is extracted, and output is given as an event. For *run off roadway* event extraction, a rule-based condition that includes a dictionary of off-road events is applied. A driver leaves a travel lane is referred to *run off roadway* events (C. Liu & Subramanian, 2009). The off-road events include bridge, cable barrier, concrete barrier, cross median, culvert, curb, ditch, embankment, fence, guardrail, fire hydrant, submerge, pole, mailbox, parked vehicle, reenter, sign post, traffic signal, trees, utility pole and construction

---

<sup>15</sup> en\_core\_web\_trf is an English transformer pipeline . the components are transformer, tagger, parser, ner, attribute\_ruler, lemmatizer. [https://spacy.io/models/en#en\\_core\\_web\\_trf](https://spacy.io/models/en#en_core_web_trf)

barrel. In case of presence of the off-road events in the sequences, the *run off roadway* event was inserted before the first off-road event in the sequence.

The result of Step 4 of the algorithm provides a list of crash events of a crash narrative in chronological order. Once Steps 1–3 are completed, Step 4 is required to generate crash sequences. As a byproduct, the algorithm extracts event related action words from narratives.

**Algorithm 1:** HGEN

```
input : Event word Dictionary  $E_d$ , Action Word Dictionary  $A_d$ , ROR dictionary  $ROR_d$ , Crash Narrative  $N_{rr}$ 
output: Event Sequences  $E_{seq}$ 

1 begin
2   Doc = convertNarrativeToToken( $N_{rr}$ )
3   t = token
4    $i_t$  = index of token t
5   e = event word in  $E_d$ 
6    $Ws_{key}$  = event related words for event word e in  $E_d$ 
7    $w_s$  = event related word in the event words  $Ws_{key}$ 
8   tempSeq = temporary sequence holder
9   for  $i, t$  in Doc do
10    for  $e, Ws_{key}$  in  $E_d$  do
11      for  $w_s$  in  $Ws_{key}$  do
12         $w_{s\_len}$  = countNumberOfWordIn  $w_s$ 
13        if  $Doc[i:i+w_{s\_len}] == w_s$  then
14          ances = 'c'
15          for  $w_t$  in  $Doc[i : i + w_{s\_len}]$  do
16            if  $w_t$  is a subject then
17              ances = 'c'
18              for w in  $w_t$  do
19                if w is a passive subject then
20                  ances = findTheClosestAncestorsOf w
21                  store e in tempSeq
22                end
23              end
24            end
25          end
26        end
27        if ances == 'c' then
28          counter = i
29          while  $Doc[counter]$  is not verb & counter > 0 do
30            counter = counter - 1
31             $t_{dic}$  = getAllActionWordsFrom  $A_d$ 
32          end
33          if  $Doc[counter]$  in  $t_{dic}$  & e is not  $e_1$  of tempSeq then
34            | store e in tempSeq
35          end
36        end
37      end
38    end
39  end
40  for e in tempSeq do
41    if e in  $ROR_d$  and  $ROR$  not in  $E_{seq}$  then
42      |  $E_{seq} = ROR + tempSeq$ 
43    end
44  end
45 end
```

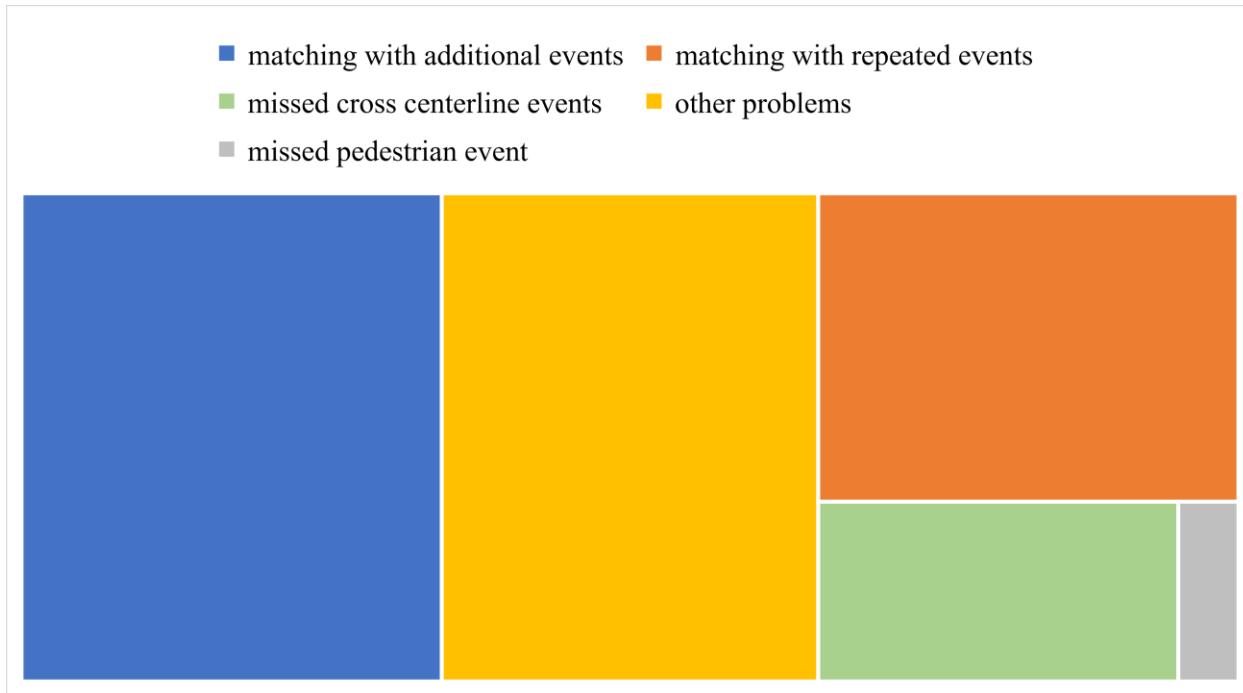
Figure 6.3-1 Pseudocode of HGEN Algorithm

## 6.4 Result Validation

The number of events in the crash narrative may be fewer or greater than that in the structured data. As there is no established method for evaluating crash sequences derived from either, a direct comparison between the two is difficult. Therefore, the validation of events is based on the temporal sequence. In addition, a manual review of 100 randomly selected crashes was performed to investigate potential causes of the disparity if there are any.

This research compared the timestamps of events generated from crash narratives with the temporal order of events in the structure data. If a crash sequence missed at least one order of events in the structure data, it is considered to be incorrect. In the case that the HGEN-generated sequences contain more events than those of structure data, only the events in both sources are compared. All sequences with fewer than two events were discarded as they were insufficient to determine the temporal patterns. The comparison result shows that the HGEN algorithm achieved an accuracy of 82.73% for all single-vehicle crashes involving fatalities and injuries. Furthermore, the accuracy was 87.37% for narratives with maximum length 500 (average narrative length is 479).

Figure 6.4-1 shows the result of a manual review of 100 of 1,638 randomly selected cases from HGEN. Sixty-four cases contain all the events with exact order recorded in structure data, including forty-five cases that are exact matches and nineteen cases that contain additional event(s). Twelve cases contain correct sequences but repeated events; six cases miss a cross centerline event. One case is coded wrong because it missed a pedestrian-related event. The remaining 17 cases have miscellaneous minor issues.



**Figure 6.4-1 Findings from Manual Evaluation**

A detailed check reveals that the reason for missing *cross centerline* event in 6 cases was because *cross centerline* was not explicitly stated in the narratives. Other issues included: the “overturn” was written as “over turn”; the misclassification of events in structure data such as embankment was recorded as ditch; a post was recorded as sign-post in structure data, but no description was given in the narrative and thus, it may be *other post*; and failure of lemmatizer of Spacy package to convert the word “fencing” into “fence”. Interestingly, the algorithm rarely failed to extract any events. Only 1 in 100 cases, the algorithm failed to extract a pedestrian-related event. After investigation, it was found that "walking" and "standing" are the most frequent pedestrian-related terms mentioned in pedestrian-related narratives. However, police officers also frequently used these terms in non-pedestrian crashes (e.g., driver was standing/walking/able to walk), leading to incorrect pedestrian event extraction.

The above validation confirmed that the algorithm can generate crash sequences from crash narratives with reasonable accuracy. It is evident from manual review that the algorithm can extract additional events from narratives that were previously missed or ignored in structure data. Thus, the proposed HGEN provides an opportunity to examine unique aspects or circumstances of a crash that are not captured in the data fields. A few events went undetected due to the limitation of Spacy pretrained tagger. With the continued improvement of Spacy and the flexibility to add custom rules, the tagger problems will be solved eventually.

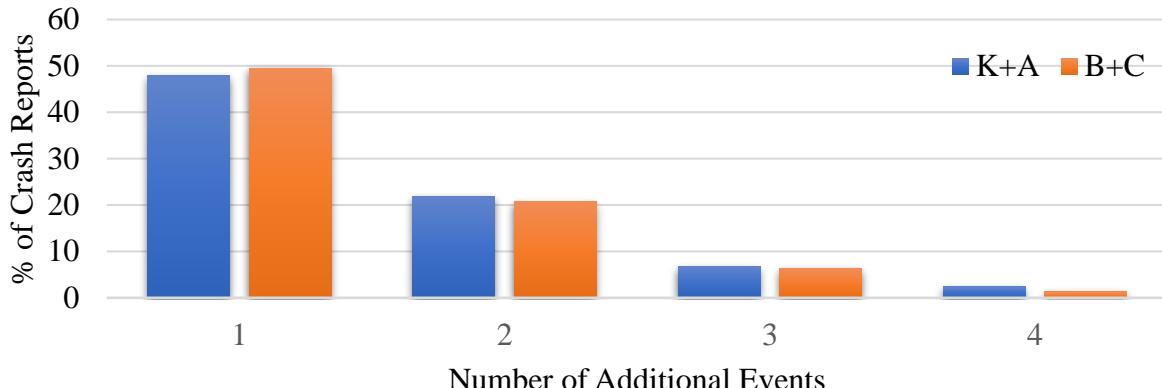
## **6.5 Application of the Results**

Complete sequence of the events is crucial to construct common scenarios associated with or leading to the most harmful event. The scenarios are often derived from the events and their sequences prior to the most harmful events so that preventative measures can be taken to mitigate and prevent these most harmful events. The proposed HGEN algorithm can be used to discover new events that are not available in the structure data fields, particularly the ones that occur prior to the most harmful event. In addition, comparing events between HGEN generated sequences and the structure data events can shed light on the quality of structure data and narratives, as well as solutions for improvement. The following section describes the application of the algorithm in traffic safety. The specific applications include extracting additional or missing events, identifying common patterns of events, and spatial analysis of the result.

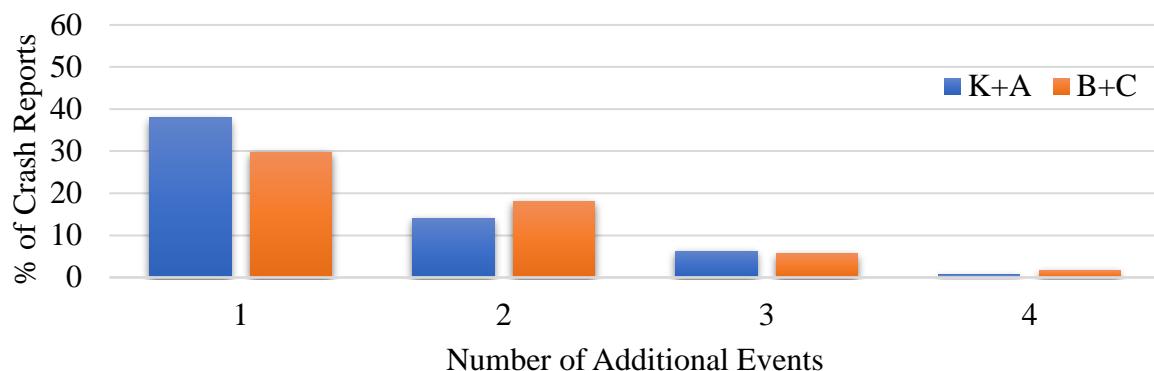
### **6.5.1 Finding Additional Events in the Crash Narrative**

Generating the sequence of events directly from a narrative can help better understand crash propagation. It was found that 78.08 % of crash narratives contain additional crash events which are not captured in the data fields. Figure 6.5-1(a) shows the statistics of additional crash events found in the HGEN generated crash sequences by level of injury severity. The majority of the narratives, including fatal (K), suspected serious injury (A), suspected minor injury (B), and possible injury (C), contain one additional event, while roughly one-fifth of the narratives contain two additional events. The number of narratives in (K+A) that contain more than one additional event is slightly greater than in (B+C). Most of those additional events in (K+A) were *run-off-roadway, ditch, tree, and overturn* events, indicating that they may occur on rural highways but somehow are missed from the data fields.

The greater the number of events that occur prior to the most harmful event, the greater the number of options available to prevent a crash. Therefore, the finding of additional events before the most harmful events is undoubtedly an important piece of information to design crash preventive measures. To do that, the HGEN generated events were compared to structured data events. Figure 6.5-1(b) shows the number of additional events found before the most harmful events. 58.81% of (K+A) narratives contain at least one additional event prior to the most harmful events, which is slightly higher than (B+C) narratives (55.25%). It implies that additional information before the most harmful events is more likely to appear in (B+C) narratives than (K+A) narratives. These additional events can help prevent crashes before the incidence of (B+C) type of injuries, which account for two thirds of the crashes.



(a)

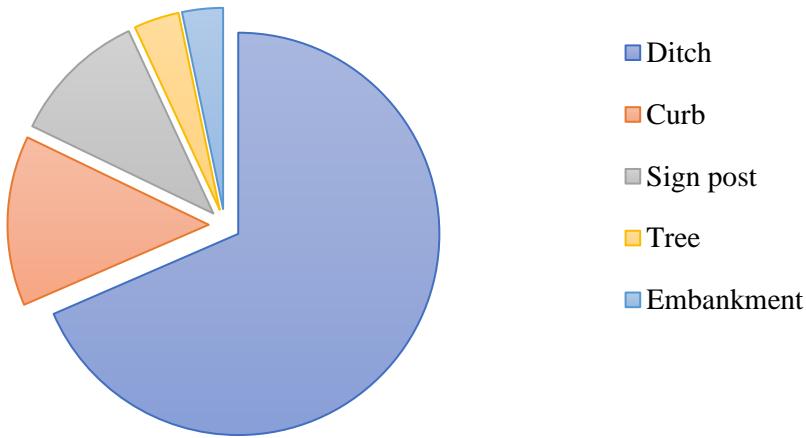


(b)

**Figure 6.5-1 Percentage of Additional Events by Injury Severity - (a) Regardless the Location of Most Harmful Events, (b) Only before the Most Harmful Events**

Further investigation revealed that *run off roadway* was the most frequent additional or missing event extracted using HGEN. Surprisingly, 653 out of 1638 (or 39.87%) cases did not contain *run off roadway* event in any field of the structure dataset. Figure 6.5.2 shows five most occurring events after *run off roadway* event that were not recorded in structure data. Alarmingly, *ditch* and *embankment* events occupied 69.07% of the total missed *run off roadway* cases. In the US, run off roadway crashes account for roughly 70% of single-vehicle fatalities,

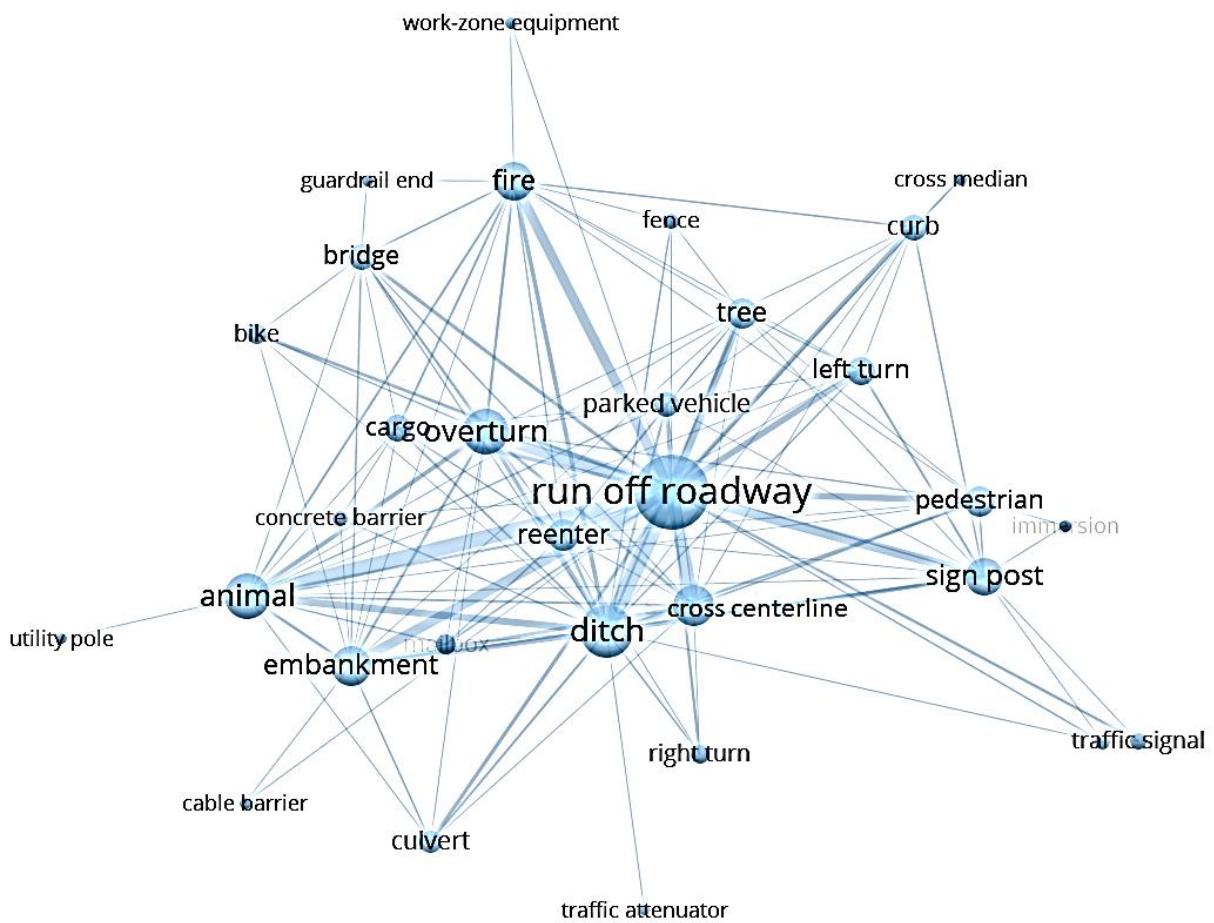
with the majority occurring in rural areas (C. Liu & Subramanian, 2009). The result indicates that the importance of *run off roadway* was not well understood by the police officers.



**Figure 6.5-2 Most Frequent Events Associated with Missed ROR in Structure Data**

The preceding discussion demonstrates that the narratives include important events such as a *run off roadway*, *ditch* and *sign post* that were frequently missed in structure data. In fact, 31 different types of events, including multiple events of a crash are omitted from structure data. Figure 6.5-3 shows the association of additional events in a two-dimensional network diagram constructed using the association method from (Van Eck & Waltman, 2010). The size of the node represents the number of occurrences of an event in different sequences of events. If an event occurs multiple times in a sequence, the method only counts it once. The width of the edge (lines) represents the number of occurrences of two events in a sequence of events; but, it is not required that the two events be adjacent to one another. The distance between two nodes represents the relatedness of two events. For example, *run off roadway* and *reenter* are located close to each other, indicating that the relations (distance) of *run off roadway* with other events

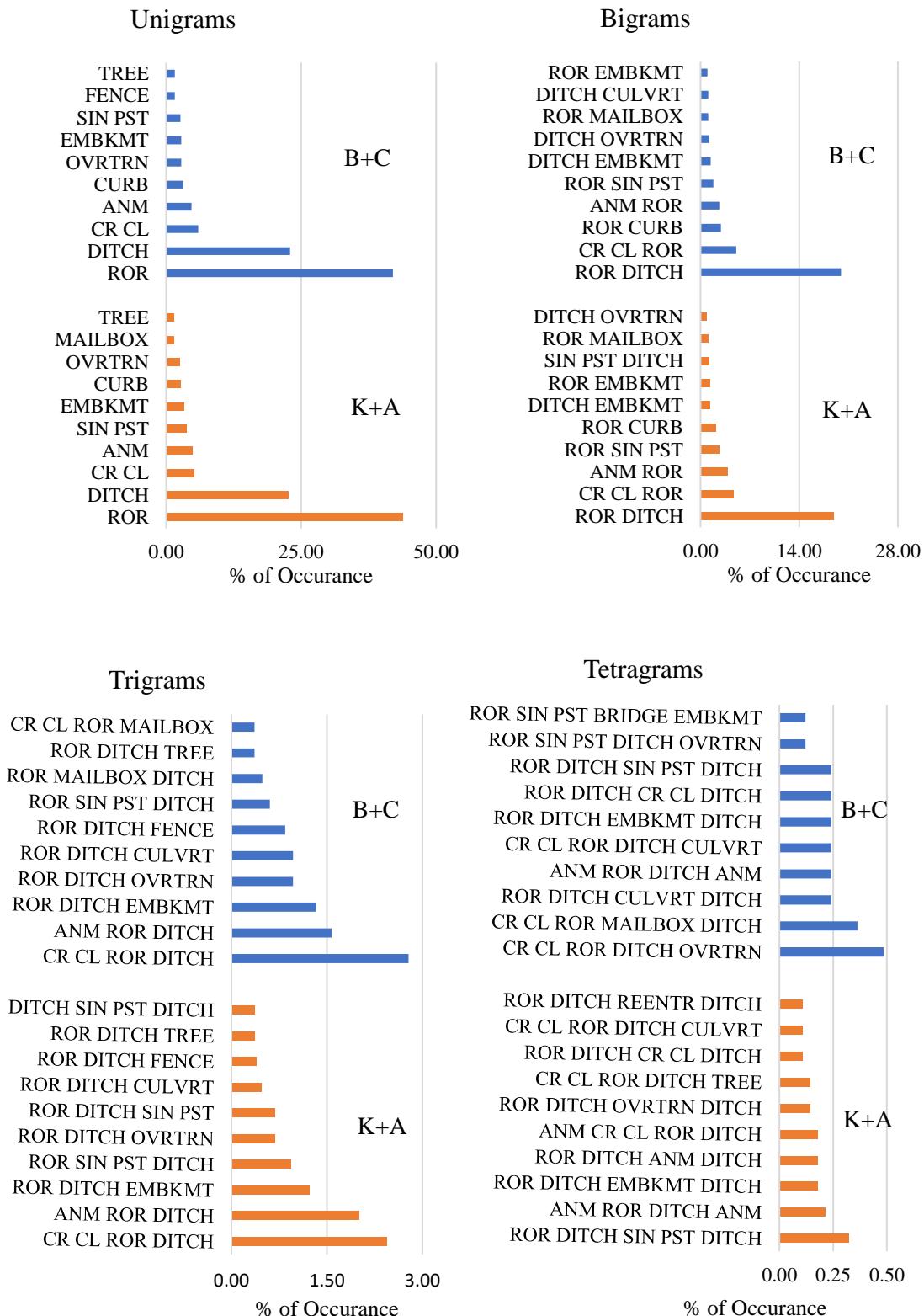
are very similar to the relations of *reenter* with those events. The location of the *run off roadway* event to the center of the network diagram indicates that the event is relatively common with other events in the crash narratives that were omitted in the structure data. Events that are near the edge and linked to multiple events indicate that the event is weakly or not related to distant events in the network diagram.



**Figure 6.5-3 Additional Events found before the Most Harmful Events in Crash Narratives**

## 6.5.2 Identify the Scenarios (Subsequence) in (K+A) and (B+C) Injury Crashes

The analysis of n-gram is one of the most popular text analytic methods because it is simple and powerful. The n-gram method looks for exact event matches in sequences and does not allow event gaps. Herein n-gram refers to the subsequence of n ordered events derived from event sequences. A variety of n-grams were generated, including unigram, bigram (pair of two consecutive events), trigram (pair of three consecutive events), and tetragram (pair of four consecutive events). Overall, the most common subsequences prior to the most harmful events are *run off roadway* (ROR), *run off roadway ditch* (ROR DITCH), *cross centerline run off roadway ditch* (CR CL ROR DITCH), and *run off roadway ditch sign post ditch* (ROR DITCH SIN PST DITCH). Figure 6.5-4 shows that the most common events are *run off roadway* (ROR), *ditch* (DITCH) and *cross centerline* (CR CL). 9 out of the top 10 bigrams are common between K+A and B+C. The *ditch culvert* (DITCH CULVRT) in B+C and *sign post ditch* (SIN PST DITCH) in K+A were exclusively found. In K+A, the *run off roadway ditch* occurred 19.03% of the time. It occurred at the beginning of the K+A crash sequences 71.51% of the times for all the *run off roadway ditch* and 14.23 % of the times for all the bigrams of K+A. In B+C, it occurred 18.97 % of the times. with the highest occurrence rate (68.68 % of the 18.97 %, and 13.03% of all the B+C bigrams), occurring at the beginning of the B+C crash sequences. Interestingly, the *run off roadway ditch* was also located at the topmost trigram *cross centerline run off roadway ditch* (CR CL ROR DITCH) and the topmost tetragram *run off roadway ditch sign post ditch* (ROR DITCH SIN PST DICTH) in K+A.



**Figure 6.5-4 Most Frequent Subsequences in K+A and B+C Crash Sequences**

Therefore, the most critical subsequences of events are *run off roadway ditch* and *cross centerline run off roadway ditch*, and in both K+A and B+C, including *run off roadway ditch sign post ditch exclusively in K+A and cross centerline run off roadway ditch over turn in B+C*.

Further investigation into the events in relation to crash subsequences revealed that the most harmful events are often occurred in the ditch, which is usually located outside of the city. No traffic barrier or curb was found in sequence of events when ditch was the most harmful events. In addition to *ditch*, the common most harmful events associated with the most common crash subsequences in the sequence of crash events are *over turn*, *tree*, *culvert*, *utility pole* and *light pole*.

The relationship of the most harmful events with their subsequences of events can help to determine which subsequence is important to prevent the most harmful event. The relationships are captured in two network diagrams (Figure 6.5-5 & 6). Figure 6.5-5 shows the relationship between the most harmful events and the subsequences of events for K+A crashes. The association rule yielded five clusters of events. The clusters are dominated by the three most harmful events *ditch* (Cluster 2), *tree* (Cluster 4), and *overtur*n (Cluster 5), two subsequences *run off roadway* (Cluster 1) and *run off roadway ditch* (Cluster 4); and a few smaller ones (Cluster 3). Clusters 2 and 5 are far apart, indicating that they have little in common, and many of their subsequences are identical. The following are the specific details for each cluster:

*Cluster 1:* A cluster in which *run off roadway* is the dominant subsequence and is associated with several most harmful events, such as *concrete barrier*, *mailbox*, *curb* and *guard rail end*. It implies that treating the subsequence *run off roadway* can prevent those most harmful

events from occurring. It is also strongly associated with *tree* (Cluster 3) and *ditch* (Cluster 5) outside of its own cluster, indicating its significance in preventing other pattern of crashes.

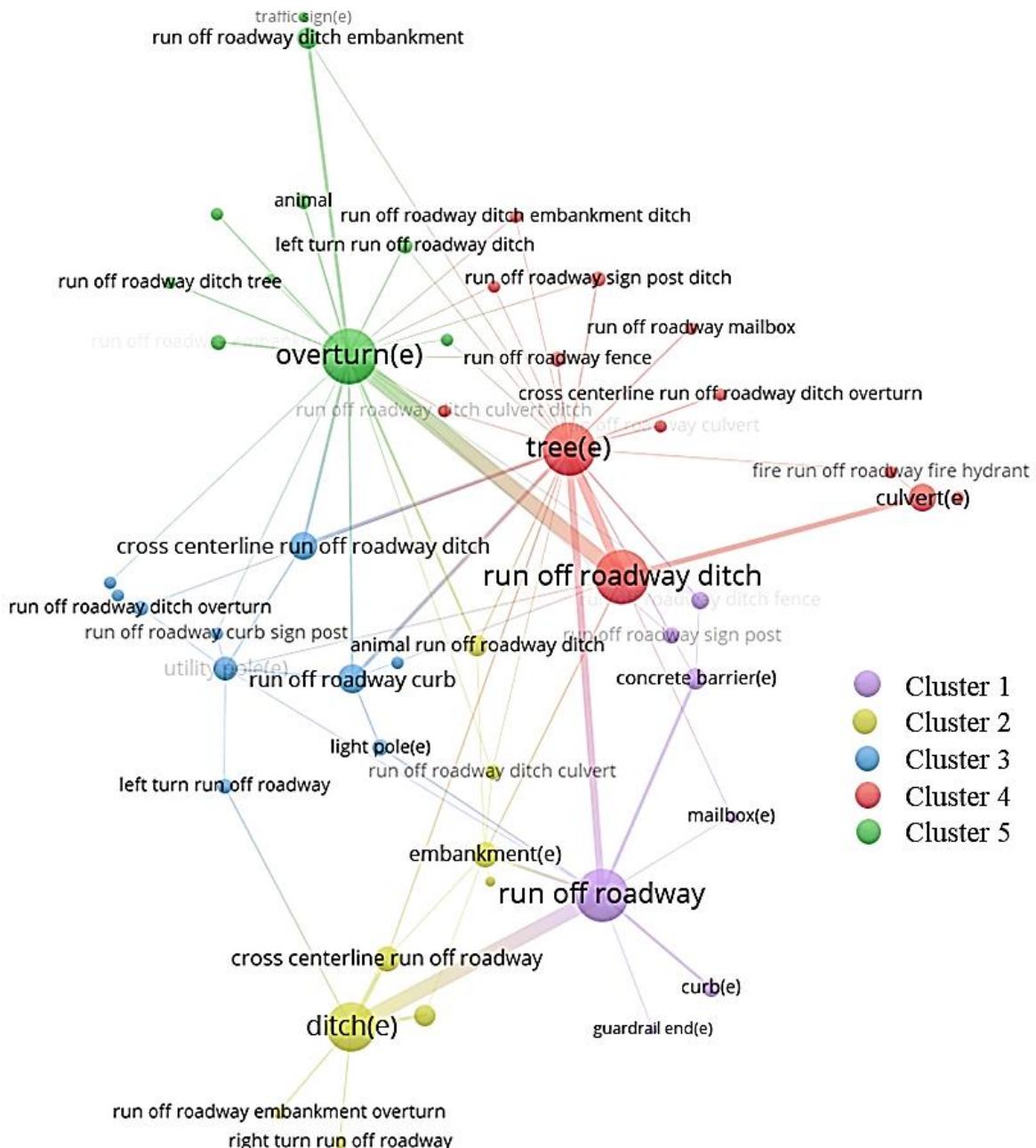
Cluster 2: This cluster contains two of the most harmful events, *ditch* and *embankment*, with *ditch* being the most alarming. In comparison to cluster 1, the subsequences associated with these two events contained more events. It implies that more alternatives (events) are available to prevent the most harmful events.

Cluster 3: It is a small cluster containing two other most harmful events *utility pole* and *light pole*. Most of the subsequences contained on-road events such as *curb*, *left turn* and *run off roadway*, and there were few events in the subsequences to control these two most harmful events. However, only a few occurrences of these two subsequences were observed in K+A, indicating that their impact is less significant for immediate concern.

Cluster 4: This is the most dominant cluster, connecting all the others through the most harmful events and subsequences. The *tree* and *runoff roadway ditch* are the most dominant within their own cluster and are also largely associated. The subsequence *run off roadway ditch* is also strongly linked to a most harmful event *overturn*, indicating that *run off roadway ditch* has a significant impact on *tree* and *overturn* related crashes.

Cluster 5: As can be seen, *overturn* is the most significant event that connected all the subsequences within its own cluster and associated with many events in cluster 2, 3 and 5. Most of the subsequences contained multiple events, indicating that there are many options to prevent *overturn*-related crashes.

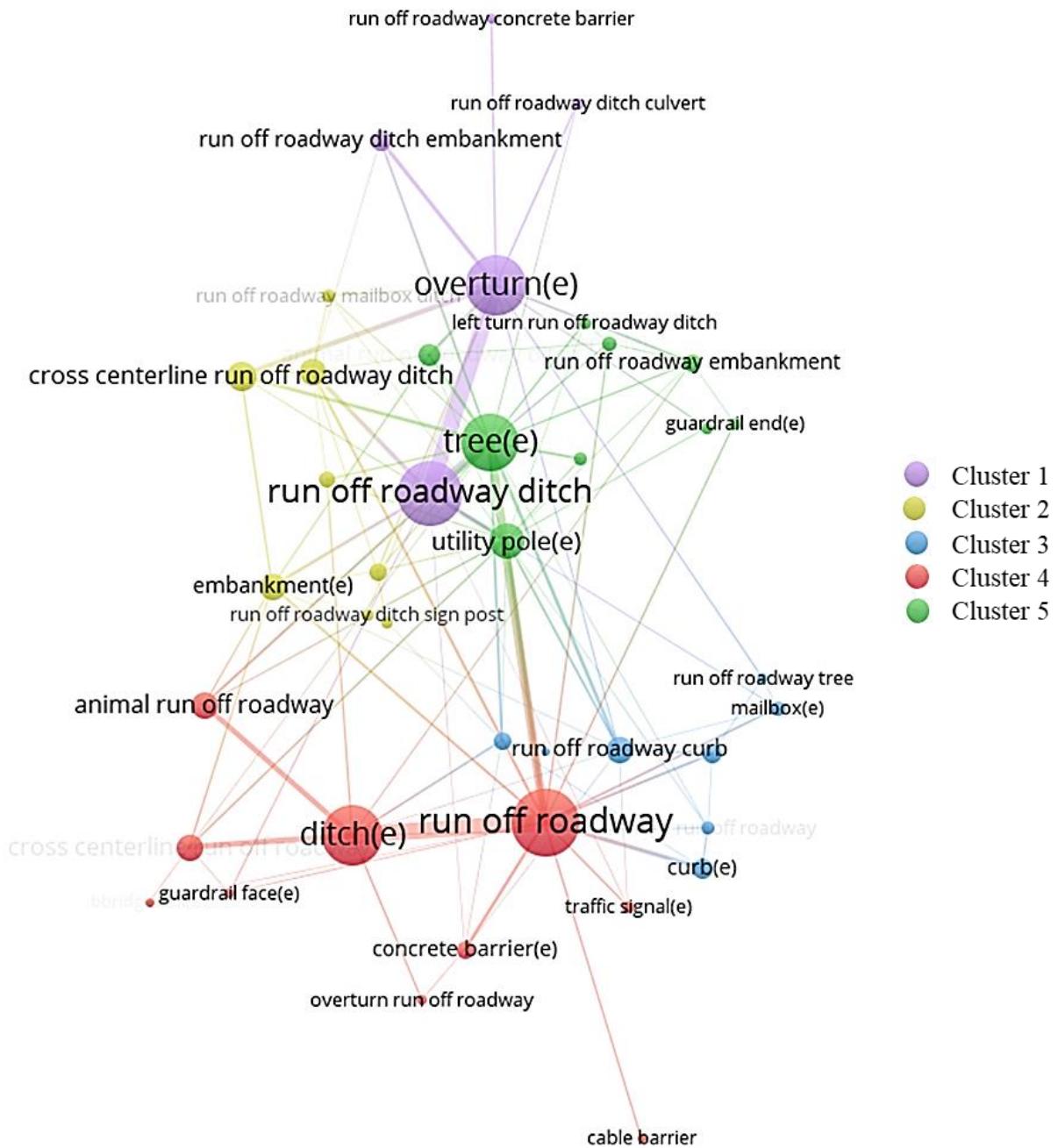
Overall, most of the subsequences associated with most harmful events *overturn*, *ditch* and *tree* contained multiple events. The *run off roadway* and *run off roadway ditch* are the most important events or event subsequences that can be considered when taking preventive action.



**Figure 6.5-5 Association of Most Harmful Event with the Subsequence of Events Occurred before the Most Harmful Events (“e” in parenthesis) in K+A.**

Figure 6.5-6 shows the relationship between the most harmful events and the subsequence of events for B+C. The association rule also yields 5 clusters of events and event subsequences. The diagram can be understood as described for Figure 6.5-5. For example, the clusters are dominated by the three most harmful events, including *overturn*, *tree*, *ditch* and *utility pole*, and the two subsequences, including *run off roadway ditch* and *run off roadway*. The findings are similar to K+A crashes, which implies that safety practitioner should consider these events and their subsequences to prevent crashes. The closeness of the *tree*, *runoff roadway ditch*, and *utility pole* suggests that they share a close relationship with their common subsequences of events.

In summary, the most crucial piece of information for preventing crashes is the subsequence of events or the most harmful events that are relatively equally distributed from others and connected to the majority of clusters. In this regard, the *runoff roadway ditch* in both K+A and B+C and the *tree* and *utility pole* in only B+C are the most significant subsequence of events and the most harmful events, respectively.



**Figure 6.5-6 Association of Most Harmful Event with the Subsequence of Events Occurred before the Most Harmful Events (“e” in parenthesis) in B+C.**

### **6.5.3 Spatial-temporal Analysis of the Sequence of Crash Events.**

A technique for obtaining narratives with additional events would save safety practitioners a great amount of time. Any pattern in narratives that correlates with any variables in structure data can provide an opportunity to find those narratives. Due to the fact that police officers write the narratives, it was hypothesized that narratives can vary significantly by geographic location, highway class, and jurisdictions. To investigate the possible hidden patterns in narratives, 11 variables were selected from structure data (Table 6.5-1) to determine the most significant variable and its relationship to the narrative.

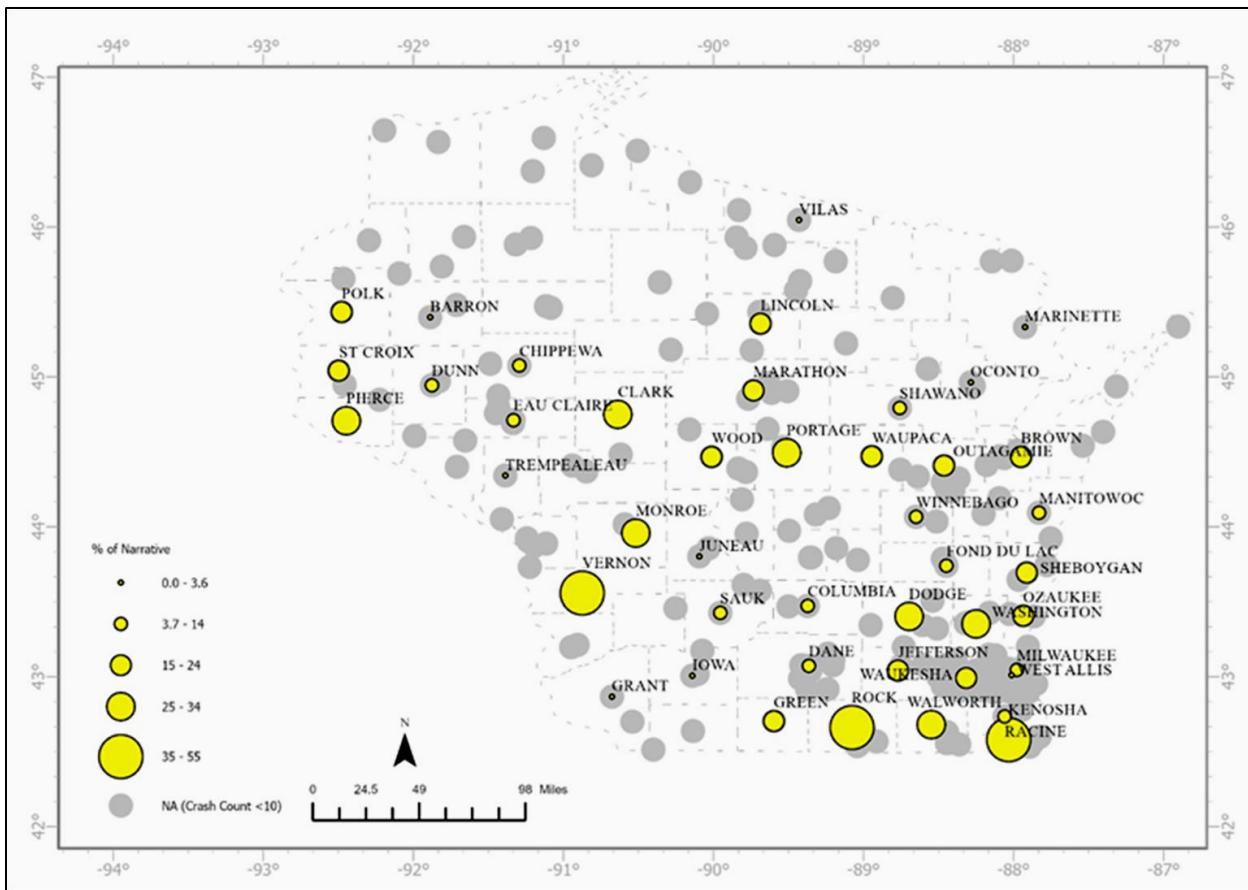
A chi-square test was conducted to determine whether a correlation exists between the presence of additional events in narrative and the selected variables. Chi-square test is used in statistics to test the independence of two events and to select the features which are highly dependent on the response. The greater the Chi-Square value, the more dependent the feature is on the response. Table 6.5-1 shows that the location of the reporting law enforcement agency (JRSDTN) is highly associated with the presence of additional event in the narratives.

**Table 6.5-1 Variables Associated with Additional Events**

Variables	Description	Chi-Square
CNTYNAME	The name of the county in which the crash occurred	5.52
MUNINAME	The name of the municipality in which a crash occurred	14.56
HWYCLASS	A code which describes the type of road the crash took place on	0.23
INJSVR	The severity of a crash based on the most severe injury to any person involved in the crash	0.08
JRSDTN	Text describing the location of the reporting law enforcement agency	44.30
AGCYTYPE	The type of law enforcement agency that reported the crash	8.69
SEX1	The sex of a person involved in a crash	0.49
INJSVR1	The severity of a crash based on the most severe injury to any person involved in the crash	0.47
DAYNMBR	The day of the week on which a crash occurred	0.34

CRSHMTH	The month in which a crash occurred	3.20
CRSHSVR	A code describing the overall severity of a crash	0.0002

In Figure 6.5-7, the percentage of narrative contained additional event information has been plotted to investigate the spatial distribution of the location of the reporting law enforcement agencies. In the figure, the location of jurisdiction area is the centroid of all the single crashes occurred within the respective jurisdiction area. The jurisdiction areas that had less than 10 crashes (black colored point on map) were discarded from this analysis. It was observed that the narratives recorded in the south-eastern jurisdiction areas, which include one of the largest cities in Wisconsin, contained more additional event information.



**Figure 6.5-7 Approximate Jurisdictional Location of Crash Report Officers and the Rate of Crash Report Contained Additional Events**

## 6.6 Conclusion

This research initiated a new research direction in the field of transportation text analytics by devising a method to generate sequences of crash events from fatal and injury-related single vehicle crash narratives. To develop the algorithm, this study investigated and used a variety of natural language processing techniques. Furthermore, the findings were discussed from various perspectives in order to provide varying insight into the crash narratives and the potential application of the results.

To generate sequences of events from narratives, this study developed four algorithms: part-of-speech tagging (PT), pattern matching with part-of-speech tagging (PMPT), dependency parser (DP), and hybrid generalized (HGEN). HGEN generated the most accurate crash sequences when compared to other methods. It uses predefined event and event-related action words that were derived from crash narratives. The algorithm was developed, tested, and validated using injury and fatal-related single vehicle crash data collected from Wisconsin Department of Transportation (WisDOT). This research utilized Python package Spacy for NLP pipeline and a part of NoisyOR method for text mining. To examine the results, an automatic validation procedure and a manual review were conducted. First, the temporal order of events in the sequence was validated for the entire dataset, and then 100 randomly selected outcomes of the HGEN were manually reviewed to validate the correctness in event sequence. HGEN achieved accuracy of 82.73 % for entire dataset and 87.47% for narratives with a maximum length of 500 characters in terms of temporal order of events.

The results provided valuable insights about the structure data and crash narratives in terms of missing and additional events. To discover patterns in the HGEN generated crash sequence, this study utilized n-gram method that looks for exact event matches in sequences and do not allow event gaps. The investigation revealed that the most critical and longest subsequence of events are *run off road way ditch* in both K+A and B+C, respectively. The common most harmful events are *ditch, over turn, tree, culvert, utility pole* and *light pole*. Most of the subsequences associated with the most harmful events consist of multiple events. Thus, the subsequences can help to prevent common fatal and injury-related crashes. Additionally, research revealed that in k+A, *overturn, tree*, and *ditch* are the three most harmful events, and

*runoff roadway ditch* and *runoff roadway* are the two subsequences. In B+C, the two subsequences are *run off roadway ditch* and *run off roadway*, and the three most harmful events are *overtake*, *tree*, *ditch*, and *utility pole*. The *runoff roadway ditch* in both K+A and B+C, as well as the *tree* and *utility pole* only in B+C, are the most significant series of events and the most harmful events, respectively.

The investigation also revealed that some events (e.g., *run off roadway* and *cross centerline*) were not mentioned explicitly in the narratives, and some narratives provided no indication of the events. For example, most *animal* event-related narratives did not specify whether the animals were alive or dead at the time of the crashes. Due to the complexity of the scenarios, these events demand a separate research study. Furthermore, some narratives are noisier than others because of repeated information from the driver, witnesses, and police officers, resulting in several redundant events. Finally, since this analysis only uses data on fatal and injury-related single vehicle crashes, the findings of this research may not be generalized to the entirety of police-reported crashes.

# **Chapter 7. DEVELOP A CRASH INFORMATION EXTRACTION, ANALYSIS AND CLASSIFICATION TOOL (CIEACT) USING TEXT MINING TECHNIQUES**

## **7.1 Introduction**

Crash report is the primary source of data for analyzing a crash and identifying contributing factors. When a crash is reported, the law enforcement agency records crash information in appropriate fields of the structure data, and crash narratives. The narrative describes the sequence of events for all units involved in a crash and records additional information regarding citations, witnesses, drugs/medication, hazardous materials spillage from trucks and buses, trailer and towed, school bus information, etc. Information captured in the crash narrative is crucial, as it captures the unique and varying circumstances of each crash scene. These information are particularly helpful when analyzing misclassified or overlooked crashes.

Safety professionals mainly rely on the manual work of sifting through each narrative to extract relevant crash information. The manual review process is time-consuming and labor intensive. Additionally, the quality of a manual review is inconsistent, as it is subject to the reviewer's experience and judgement. In recent years, text mining and machine learning techniques have proven to be an efficient and effective method for automatically extracting crucial information from crash narratives. However, there is currently no tool available that would allow safety practitioners to utilize these techniques for crash narrative analysis.

In this research, an online Crash Information Extraction, Analysis and Classification Tool (CIEACT) was developed that integrated the NoisyOR algorithm (Sayed et al., 2021) to develop crash classification models. A user can either use the default pretrained model or train his/her own model for crash classification. The tool presents results in tabular form and on an interactive crash map for safety analysis in spatial context. It also provides an opportunity to download the intermediate results of the user trained model. Furthermore, compared to the traditional manual approach, this tool can substantially enhance crash report review quality and efficiency.

The following section describes the functionalities, framework, and system architecture of the tool. Next, the Backend of the tool, which includes the classification algorithm, database, and web framework has been discussed. Then a detailed on the Frontend of the CIEACT, followed by testing and evaluation of the tool has been given. After that a case study on secondary crash classification was conducted for the user trained model. Finally, the limitation and the direction of future development of the tool has been described in the conclusions and recommendations.

## 7.2 Functionalities

The functionalities of CIEACT were identified from the perspective of users, which served as the foundation for designing the framework and system architecture of the tool. In addition, the tool was designed in such a way so that it can be scaled up to manage huge online traffic and incorporate new functionalities as well as algorithms. This section discusses the functionalities of

the tool, explains CIEACT framework, and provides an overview of the system architecture of the tool.

The user survey results from “WisDOT DT4000 Crash Report Narrative Survey”, conducted in the project titled “Using Text Data from the DT4000 to Enhance Crash Analysis”, were used to determine the functionalities of CIEACT. Besides, Relevant web-based applications were considered before the functionalities of the tool were finalized. The following three points explain the core functionalities of CIEACT.

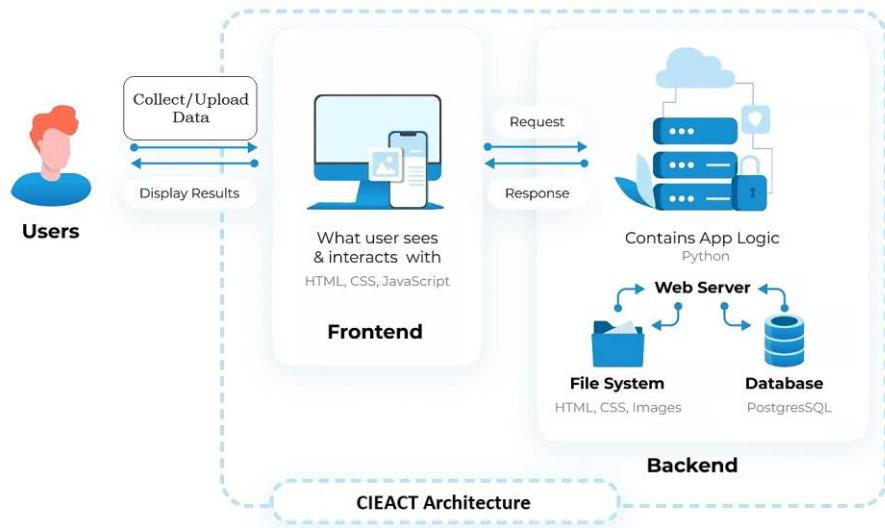
- 1) *Data Management*: The tool provides users with the capability to upload data for analysis in the form of a csv file. Keeping user data privacy in mind, all data uploaded by users is stored in temporary memory. As a result, when the user exits the tool, all data provided by the user is deleted. The user may feel the need to save their results and return later to use it. To provide this functionality, the data should be stored in a database (e.g., PostgreSQL). Although the current version of the tool does not provide this function, a pipeline to connect database and handle user data has already been developed in CIEACT. For pretrained model, all information is handled in PostgreSQL, a relational database system, which provides easy access to upload and update any pretrained model.
- 2) *Crash Classification*: CIEACT provides two options: (1) default model, which refers to pretrained model that can be used to classify any text data and (2) train model, which refers to developing model in real time using the provided *algorithms*. To use the default model, users need only upload text data (e.g., crash narrative) for analysis and select among the available default models. On the other hand, to train a model on the go, users first need to upload training data and select the appropriate classification algorithm. The

tool redirects the users to a model summary page once the model training is complete. The summary provides information about the top unigrams, bigrams and evaluation results based on different cut-off values. It helps users decide whether to move forward with the trained model or retrain their model with a new or modified dataset. The users can select an appropriate cut-off value and use their trained models to classify their test data.

- 3) *Model summary, Output and Visualization:* The CIEACT provides summary statistics of the user-trained model and presents the classification results in tabular form. Users can also visualize the locations of all crashes on an interactive crash map for spatial analysis. Finally, it allows users to download the results, which include words with probability scores for unigram and bigram, as well as narratives with classification scores for further review and analysis.

### 7.3 System Architecture

The system architecture of CIEACT is very similar to a web application architecture, which consists of user interface, webserver, database, file system and the core logics of the application (Figure 7.3-1).



**Figure 7.3-1 System Architecture (Deremuk, 2021)**

The tool has two main parts which are bounded by the dashed line. The user side, or the Frontend, is the part that is accessible by users on an electronic device such as a desktop computer, cell phone, laptop, etc. The Frontend connects various web pages of the tool that allow the user to navigate the tool. The frontend allows users to upload, classify, and analyze their data, as well as conduct spatial analysis on a map and download analysis results. The core part of the Backend of CIEACT is the Django web server, PostgreSQL database and the classification algorithms. Backend manages all the workflows of the tool, including efficient data flow between user and server, data storage and memory management, running the algorithms and result visualization. The next two sections explain the front end and back end of CIEACT in detail.

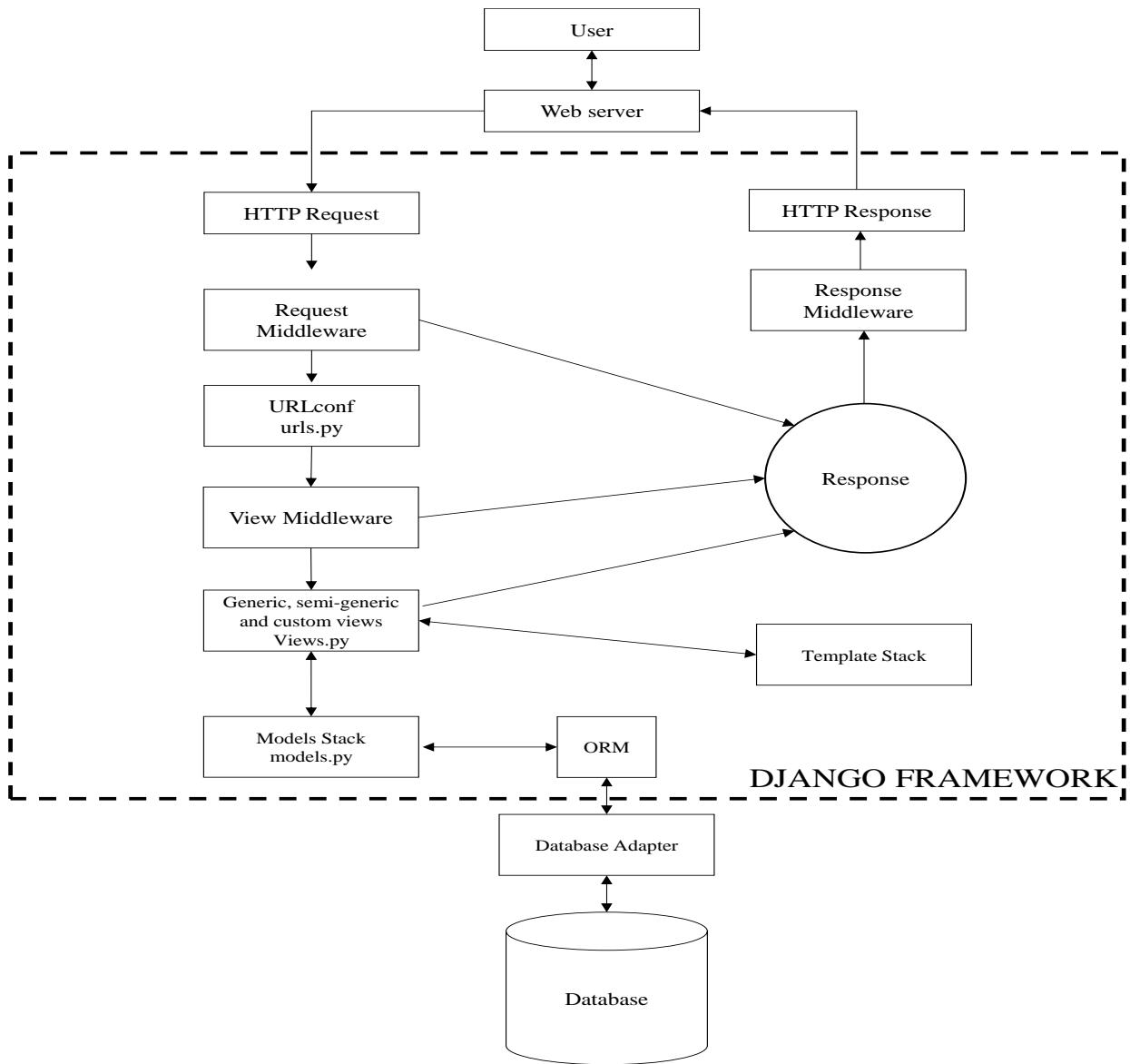
### **7.3.1 Backend**

The efficiency and user experience of the tool depend on the proper functioning of the Backend. This section describes the classification algorithm, database management system, and web framework of the CIEACT.

#### *7.3.1.1 Web Framework*

Django-based web framework follows Model-View-Controller (MVC) paradigm. The model controls the data, whereas the view defines how to display the data. Finally, controller mediates between the two and enables the user to request and manipulate the data. However, Django does it in a slightly different manner. Models in Django deal solely with data passing into and out of a database. The models work in a fashion similar to other web frameworks, but views in Django work like the controller aspects of MVC. The views are Python functions, which tie together the model layer and the presentation layer (which consists of HTML and Django's template language). In other words, Django splits the presentation layer in twain with a view defining what data to display from the model and a template defining the final representation of that information. As for the controller, the Django framework itself serves as a type of controller by providing mechanisms to determine what view and template are used to respond to a given request (For details, readers are referred to Django documentation). Figure 7.3-2 presents the overall architecture of the web-based application, with the black dashed rectangle representing the Django component. The HTTP request generated by the users via the web server goes to the Request middleware layer. It is then dispatched based on URLconf patterns to the appropriate

View. In the next step, Views perform the core part of the work which requires the use of models and/or templates to create a response. Lastly, the response goes through one more layer of middleware that performs any final processing before returning the HTTP response back to the web server. The response is then forwarded to the user via the web server. Figure 7.3-3 shows the logical diagram of CIEACT interacting with different Views, Algorithms, and Templates. Table 7.31 discusses the functionalities of the elements of logical diagrams.



**Figure 7.3-2 Overview of the Web Framework for CIEACT**

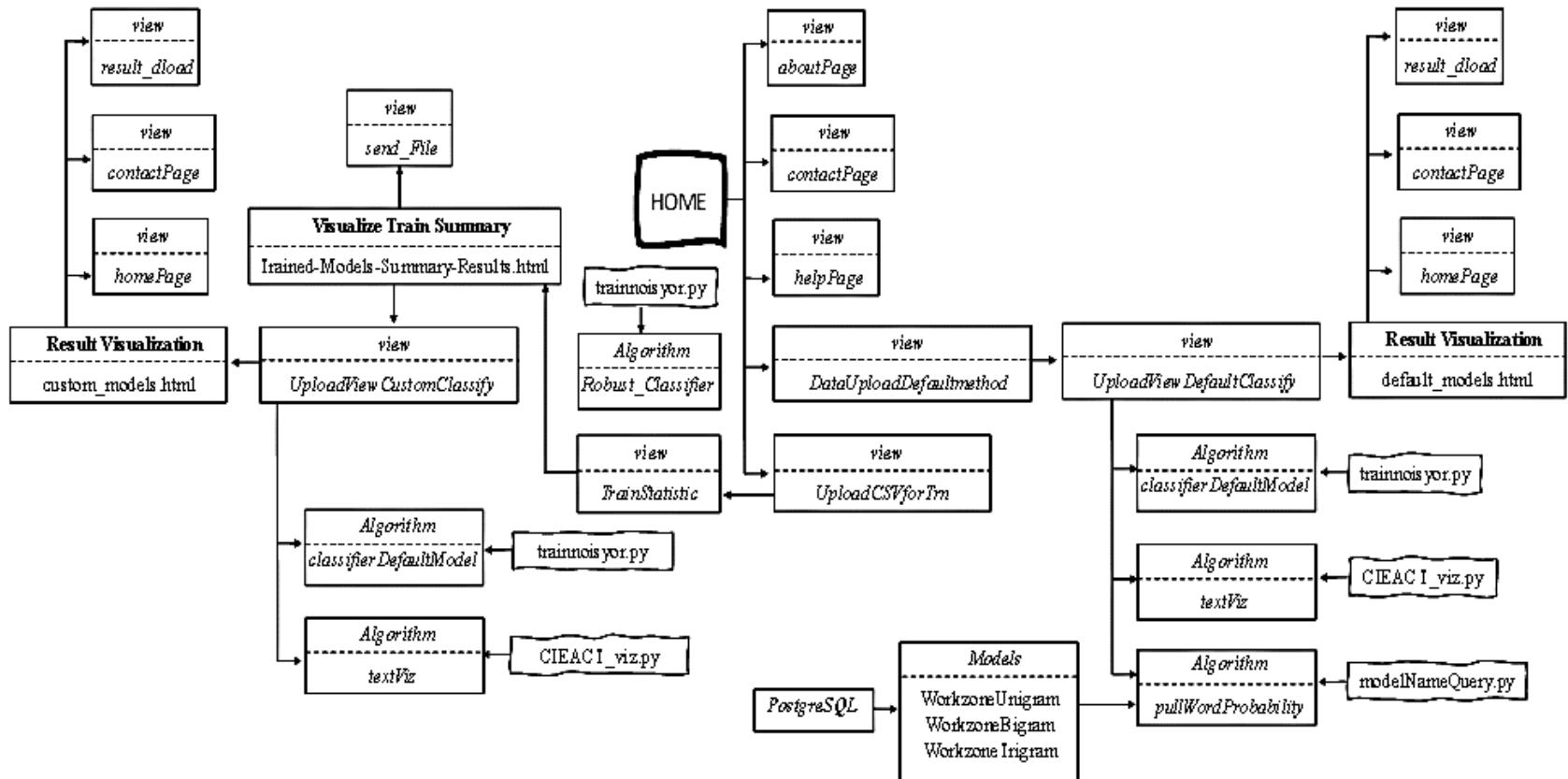


Figure 7.3-3 Logical Diagram of Functionalities of CIEACT

**Table 7.3-1 Descriptions of the Views and Algorithm in the Logical Diagram**

<b>Views</b>	<b>Description</b>
<i>aboutPage</i>	It provides an overview of the tool.
<i>contactPage</i>	It directs users to the project team for further help.
<i>helpPage</i>	It provides detailed information on how to use the tool.
<i>DataUploadDefaultmethod</i>	It navigates users to the default model.
<i>UploadCSVforTrn</i>	It checks and accepts the user-uploaded csv data for training a new model. If the upload is successful, the classification and evaluation metrics algorithms are executed, along with the functionality to download the word probability and proceed to the subsequent step for text classification.
<i>UploadViewDefaultClassify</i>	It accepts the csv data uploaded by the user for the default model. It then executes the classification and visualization algorithms. In addition, it provides the option to download the word probability results.
<i>UploadViewCustomClassify</i>	It accepts csv data uploaded by the user for their trained model. The classification and visualization algorithms are subsequently executed. In addition, it offers the ability to download classification results.
<i>wordProbability_dload</i>	It provides download functionality for word probability data.
<i>result_dload</i>	It provides download functionality for classification results.
<b>Algorithms</b>	<b>Description</b>
<i>classifierDefaultModel</i>	It imports and runs the algorithm of the default model from <i>trainnoisyor</i> to classify text narratives.
<i>textViz</i>	It provides functionalities for visualizing results in tabular format imported from <i>CIEACT_viz</i> .
<i>pullWordProbability</i>	It provides functionality for acquiring the word probability of the default model from the database imported from <i>ModelNameQueary</i> .
<i>Robust_Classifier</i>	It provides classification, evaluation matrices and word probability algorithms imported from <i>trainnoisyor</i> .
<i>classifierCustomModel</i>	It provides the functionality of a user-trained custom classifier imported from trainnoisyor.

### 7.3.1.2 *Database*

The primary database system of CIEACT is the PostgreSQL database that complies with SQL (Structured Query Language). Through the Django Model module, all necessary PostgreSQL tables are generated. The information of the default model such as words with their probability scores are stored in this database. The database is configured to store numeric information with six decimal places. To update any pretrained models, it is sufficient to simply replace the existing data of the model in the database using the standard PostgreSQL scripts.

```
y "CIEACTapp_workzoneunigram" from '<directory>\unigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
y "CIEACTapp_workzonebigram" from '<directory>\bigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
y "CIEACTapp_workzonetrigram" from '<directory>\trigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
```

```
y "CIEACTapp_workzoneunigram" from '<directory>\unigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
y "CIEACTapp_workzonebigram" from '<directory>\bigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
y "CIEACTapp_workzonetrigram" from '<directory>\trigram_probability_0.25_WZ.csv'DELIMITER ',' CSV header;
```

**Figure 7.3-4 SQL Script to Upload the Word Probability csv Data File (to be used with the Default Model)**

### 7.3.1.3 *Classification Algorithm*

The current version of the tool utilized a NoisyOR based classification algorithm developed in (Sayed et al., 2021) to classify crash narratives. The NoisyOR method calculates the probabilities of specific types of crash narratives by combining the probability scores of unigrams (words) and bigrams (two consecutive words) in the narrative. NoisyOR is basically a probabilistic extension of logical “or” (Onisko et al., 2001; Vomlel, 2006). If an input has a high

probability score (such as a value close to 1), then the combined probability in NoisyOR becomes high. The combined probability in NoisyOR is higher when more input probabilities are high. To apply the NoisyOR classifier to crash narratives, the algorithm computes probabilities of unigrams and bigrams and then combines these probabilities using the NoisyOR method to calculate the final classification score. The probability for unigram and bigram is calculated by Equation 1.

$$\text{Probability Score } (w) = \frac{\text{Positive Count}(w)+1}{\text{Positive Count } (w)+\text{Negative Count}(w)+2} \quad (1)$$

where  $w$  is a unigram, or a bigram. *Positive Count* means the number of occurrences of  $w$  in the positive narratives. Similarly, the *Negative Count* indicates the number of occurrences of  $w$  in the negative narratives.

A smoothing is applied by adding one in the numerator and two in the denominator of the Equation. This simple version of Laplace smoothing assumes  $w$  occurred at least once in a positive narrative and in a negative narrative. Smoothing done in this way ensures that among the unigrams, and bigrams with zero negative counts, those with higher positive counts receive higher probability scores. Otherwise, they will receive an unrealistic probability score of 1 because they occurred in a few positive narratives and none in negative narratives.

Additionally, minimum positive count can be fixed to control some words that provide a high probability but are very unlikely to occur in positive cases (Sayed et al., 2021). In the case of words that appear in both positive and negative narratives with a very high frequency (Count), it is likely to reduce the probability of that specific word. For example, let a unigram ‘*unit*’ appears in the narratives of a specific type of crash (positive case) 110,933 times and in all the

other narratives (negative cases) that exclude that specific crash 1,000,904 times. Then according to Equation 1, the probability score for that unigram is 0.099. This indicates that the word is not relevant for the classification task. On the other hand, if a unigram/ bigram appears frequently in both positive and negative narratives but has a higher frequency in positive narratives, Equation 1 gives a good probability score to the corresponding unigram/ bigram. For example, if the unigram '*inattentive*' appears in the narratives of a specific crash 2743 times and in all the other narratives 1808 times, Equation 1 yields a probability of 0.6023, indicating that the word is relevant for the classification task.

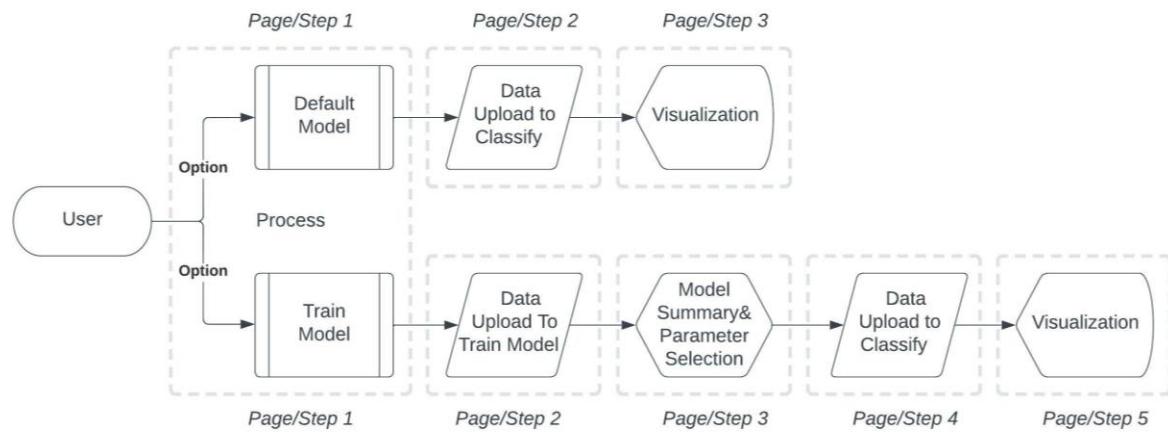
The NoisyOR score of a narrative is computed using Equation 2, where the probability score is determined by combining the probability scores of any unigrams or bigrams that appear both in the narrative and the positive words list generated by Equation 1.

$$\text{Noisy - OR Probability Score } (N) = 1 - \prod_{i,j=1}^n (1 - P_i)^j \quad (2)$$

where N is a given narrative,  $P_i$  indicates the probability score of  $i^{\text{th}}$  unigrams or bigram as computed from the training data using Equation 1, and j means the number of occurrences of that  $i^{\text{th}}$  unigram, or bigram in the crash narrative N. It should be clear from Equation 2 that if there is no unigram or bigram in a narrative with a high probability score, the probability score of the narrative will be close to zero. On the other hand, a single unigram or bigram with a high probability score will result in a high probability score for the entire narrative. Furthermore, more unigrams and bigrams with high probability scores make the combined probability score higher.

### 7.3.2 Frontend

The front end of CIEACT is the interface displayed to users on an electronic device while operating the tool. Several pages are created for navigation within the tool. This section describes the organization and functions of various web pages of the tool that are summarized in Figure 7.3-5.



**Figure 7.3-5 Overview of the Frontend of the CIEACT**

#### 7.3.2.1 Step (page) 1

The tool at first provides users with the option to select the type of work they intend to perform such as using a pretrained default model or train their own model. It also includes useful information about the tool, such as documentation, contact information to the research team, and information about the funding agency of the tool. The documentation discusses how to use the tool, such as navigation, interpreting, downloading and visualizing results.

### 7.3.2.2 Step (page) 2: “Default Model” and “Train Model” Page

<p><b>Upload Crash Data and Choose Model to Analyze Information</b></p> <p><b>Step 1: Upload Your Text Data to Classify</b></p> <p>Upload a csv file: <input type="button" value="Choose File"/> No file chosen</p> <p>Write the field names of the CSV files (* indicates required).</p> <p>*NarrID *Narrative [OFFRNARR] *0.35&lt;=Cut-off value &lt;=1 *Latitude *Longitude</p> <p><b>Step 2: Select Existing Trained Models</b></p> <p>Multiple Models can be selected at once</p> <p>NoisyOR (Unigram+Bigram) (WorkZone) NoisyOR (Unigram+Bigram) (Distracted) NoisyOR Unigram (WorkZone) NoisyOR Bigram (WorkZone)</p> <p>Email <input type="button" value="Choose an Unique ID"/></p> <p><input type="button" value="Run models"/></p>	<p><b>Upload Crash Data And Train Model</b></p> <p><b>Step 1: Upload Your Data to Train Model</b></p> <p>Upload a csv file: <input type="button" value="Choose File"/> No file chosen</p> <p>*Label *Narrative *Latitude *Longitude</p> <p><b>Step 2: Select Model to Train</b></p> <p>NoisyOR (Unigram+Bigram)</p> <p>Email <input type="button" value="Choose an Unique ID"/></p> <p><input type="button" value="Train models"/></p>
---	--

**Figure 7.3-6 The functionalities of Default Model (left) and Train Model(right)**

In this step, the user must upload their test data in csv format if they chose the 'Default Model' to classify their narratives in the previous step. Some columns are mandatory to be filled up such as "NarrID" for the crash number or any index or primary key, "Narrative" for text data, "latitude" and "longitude", "email id," and a "unique ID". The "unique ID" will be a 3-digit pin that is any text, symbol, or number. The "latitude" and "longitude" fields could be left blank in the csv data. Users will select a default classification algorithm from the dropdown menu of the model list, such as "NoisyOR (Unigram+Bigram)(WorkZone)", to classify their narratives. The user can specify a threshold value for word probability that will be used to calculate the NoisyOR score. For work zone crash classification, the default threshold value is 0.35, which

was found to be optimal in the previous study. To train a model using “train model”, users must upload their training data and choose a classification algorithm from the list of models. The field definition and functionality for “train model” are similar to the default model.

#### *7.3.2.3 Step (page) 3: Result Summary and parameter selection page of “Train Model”*

The Result Summary page shows the top 10 unigrams and bigrams with their probability scores and provides evaluation metrics for several cutoff values for training a NoisyOR classifier. The evaluation metrics can help the user to choose an appropriate cutoff value to classify their test data. The cutoff value is the minimum probability scores of the words (unigrams and bigrams) that was used to train the model. It is expected that the top unigrams and bigrams will be relevant to the target class. The trained model will perform well if the top unigrams and bigrams are relevant to the target class, and a proper cutoff value is used for the test data. This page also provides an option to download words probability results for further analysis. Figure 7.3-7 below shows a screenshot of the ‘Result Summary’ page.

#### *7.3.2.4 Step (page) 4 of Data upload Page for “Train Model”*

The purpose and functionality of this step is very similar to step 2 of Default model. However, in this step, the user do not need to select any model. They will use an appropriate cut-off value that they are selected in the previous step from the Model summary page.

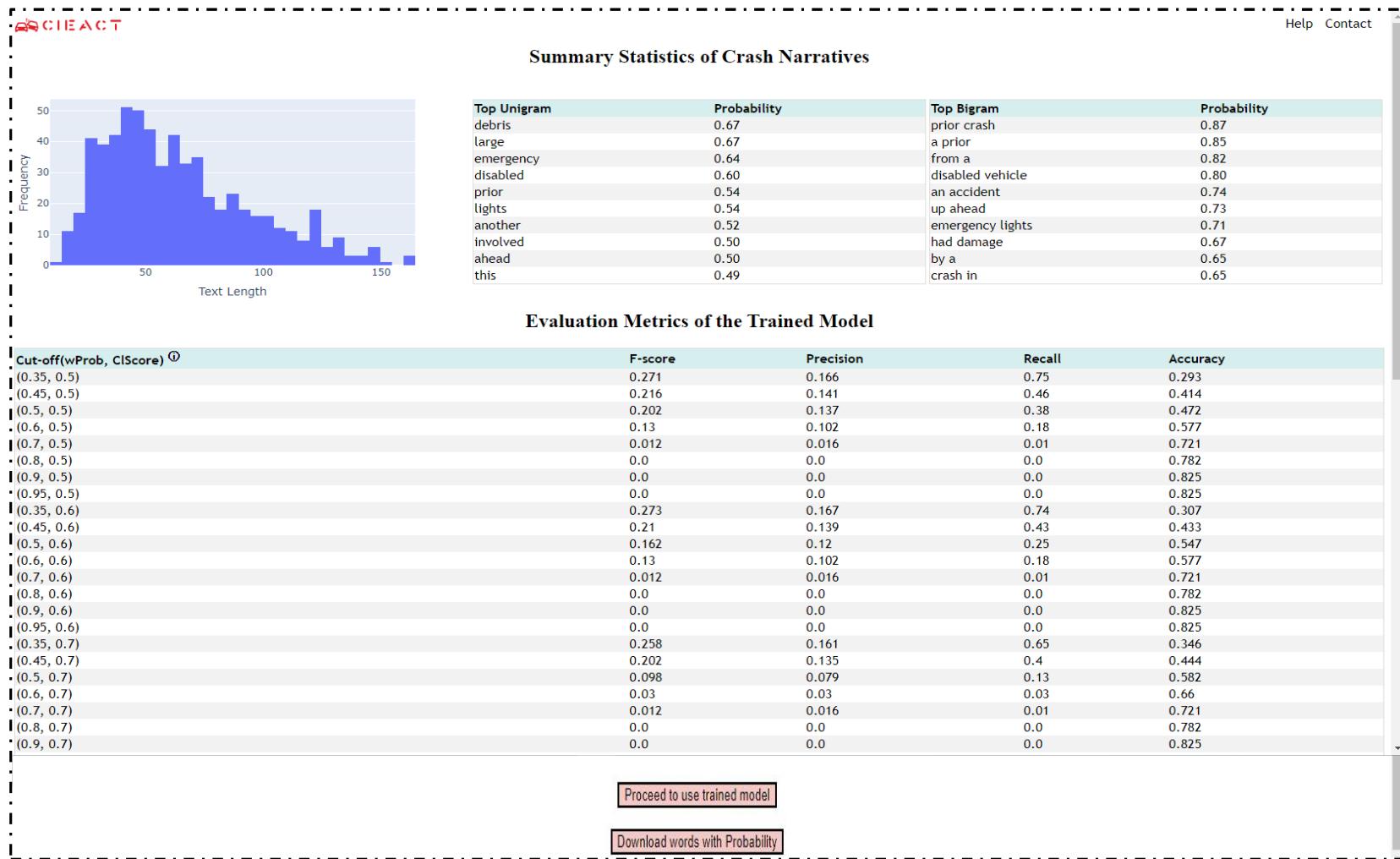
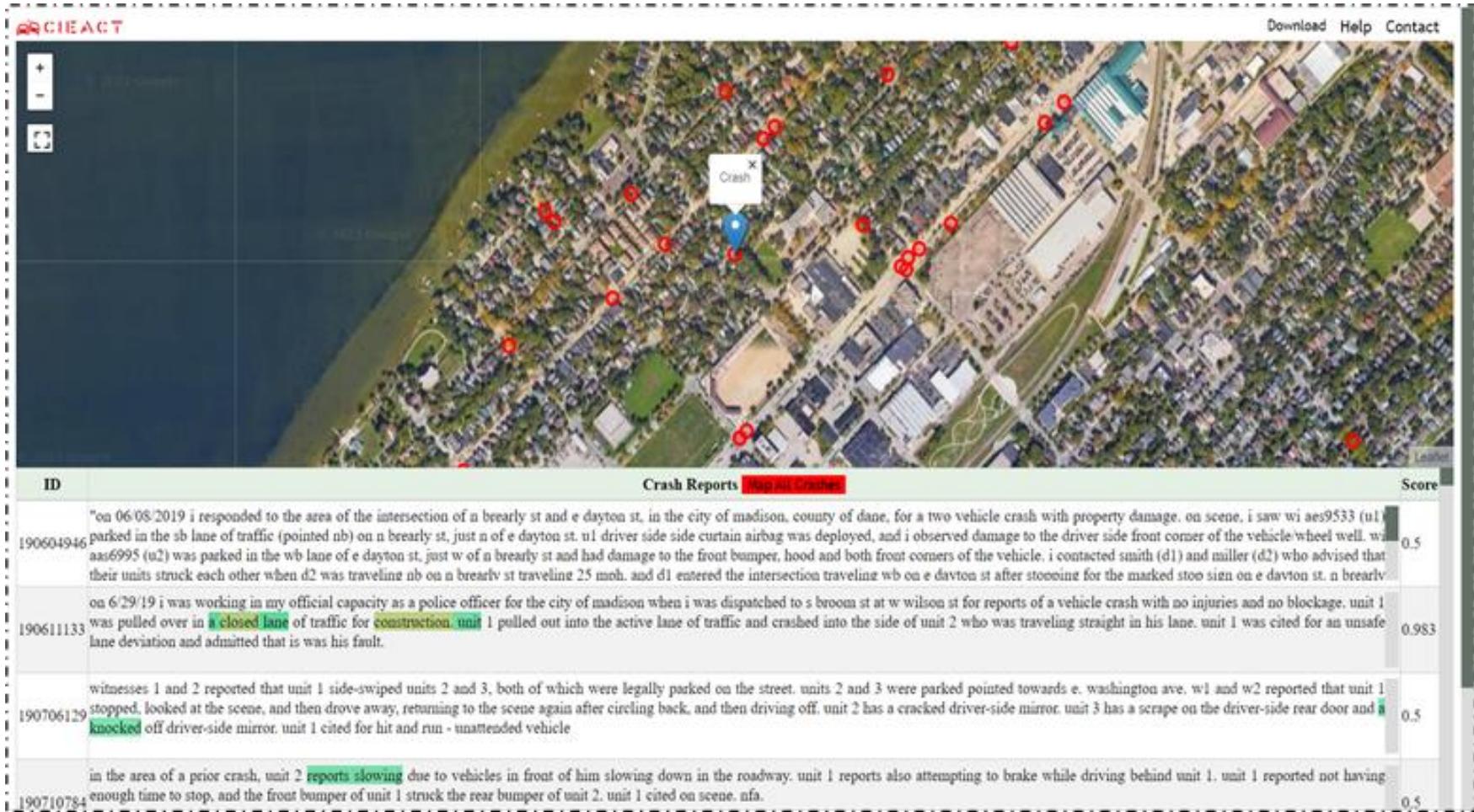


Figure 7.3-7 Screenshot Showing the Result Summary Page of CIEACT

### **7.3.2.5 Step (page) 3 of “Default Model” and Step 5 of “Train Model”**

The final step for default model and train model is same, which is result visualization and analysis. The Result Visualization page displays classification results, which includes crash ID, narratives and classification score. The tool presents the significant unigrams and bigrams from each crash narrative using color coding. A user can investigate the crash narratives quickly by viewing the tabular results. This page also provides the user with the option to see the location of any crash on a map. The map is interactive, and users can zoom in to get an idea of the surroundings of the crash location. There is also an option to visualize all crash locations on the map. These visualizations help users see the spatial distribution of crashes or find a pattern of crash locations. The navigation bar at the top right of the page provides a download option so users can download the full classification results. Figure 7.3-8 shows a screenshot of the ‘Result Visualization’ page.



**Figure 7.3-8 Screenshot Showing the Result Visualization Page of CIEACT**

## **7.4 Evaluation and Testing**

In the evaluation and testing phase, results from the default and custom trained models have been evaluated. In addition, the tool has been tested for browser compatibility, including HTML and CSS syntax validity. The following two subsections provide details on the evaluation and testing conducted on CIEACT.

### **7.4.1 Evaluation of the Output of the Tool**

To ensure proper implementation of the crash classification model in CIEACT, this study validated the classification results by comparing them with the offline model. For the user trained model, a two-step validation process was conducted. First, a NoisyOR model was trained both in CIEACT and offline (outside the tool) using a sample dataset with 10,000 crash narratives and other structured data fields. The evaluation result showed no differences in the word probability scores. Next, classification scores for the trained model in CIEACT were compared with scores for the offline model. The evaluation showed no difference in the classification scores. thus, confirming that the NoisyOR implementation in the CIEACT was successful.

In case of the default model, a single-step validation process which consisted of cross-checking the final classification results was used. This is because the tool takes word probabilities stored in the PostgreSQL database of CIEACT from a pretrained offline model. Results in CIEACT matched the results from the offline model, confirming the correct implementation of the default model in CIEACT.

#### **7.4.2 Browser Compatibility, HTML and CSS Syntax Validity Testing**

The tool was deployed on cloud using Heroku and was tested for browser compatibility using the following platforms/versions: Internet Explorer (versions 7.x and 8.x); Firefox (version 3.x); Safari (version 4.x); Opera (version 10.x); Chrome (version 5.x). The testing devices include Desktop computer; Smart phone; Tablet. The HTML and CSS syntax validity were tested on different platforms. The tool worked flawlessly across multiple browsers and devices. The tool passed all the tests.

#### **7.5 Case Study**

A case study was conducted to test the effectiveness of CIEACT in solving a new problem and how the results of the tool can be interpreted using crash narratives. A secondary crash is defined as any crash beginning with the time of detection of the primary incident where the collision occurs, either within the incident scene or within the queue, including the opposite direction, resulting from the original incident (FHWA, 2020).

Data was collected from the WisTransPortal data hub of the Wisconsin Department of Transportation through the UW-Madison TOPS lab. Positive cases were selected from the training dataset using the data field “SECDFLAG” which flagged any secondary crash in the database. An example crash narrative that is flagged as a secondary crash is

“Unit 1 was traveling northbound on i-41 in lane 1 at apple creek rd. A metal beam from the median cable barrier was laying in lane 1 of i-41 northbound from a previous crash in the median. Unit 1 drove over the metal beam, striking the undercarriage of the vehicle.

The beam became lodged under the engine of the vehicle, disabling the engine. Unit 1 moved to the right shoulder and came to rest facing north on i-41 northbound."

The narrative implies that an incidence related to a previous crash is behind the occurrence of the crash, and therefore it is classified as a secondary crash.

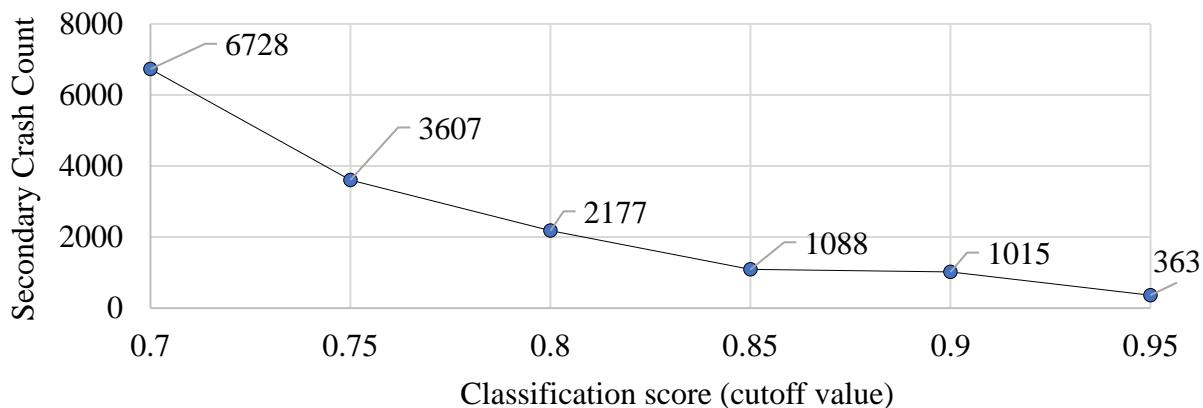
The case study used two training datasets to evaluate how the tool handles files of varying sizes and how well the classification algorithm performs in both datasets. The small dataset contained 570 negative cases, including 100 positive cases (flagged secondary crash), and the other with 5545 negative cases, including 1042 positive cases. All positive cases were taken from the secondary crashes that occurred in Wisconsin from 2018-20 in non-intersection areas of divided highways (excluding deer crashes). The negative cases were chosen randomly from a pool of 61,960 non-intersection, non-deer, divided highway crashes from 2018-20 in Wisconsin. Trained models were tested using a test dataset containing 40,516 crash narratives that occurred between January 1, 2021, and June 16, 2021.

### **7.5.1 Case Study with Small Dataset**

The model training was conducted on an ACER laptop that had Intel(R) Core (TM) i5-9300H CPU @ 2.40GHz, 2.40 GHz processor, 8.00GB of RAM and 64-bit Windows operating system. The model training was completed in approximately 3 seconds. Only nine unigrams and all bigrams had probability scores greater than 0.70 among the top 20 unigrams and bigrams (APPENDIX F). Most of the words were irrelevant to secondary crashes. The top three bigrams pertaining to secondary crashes were "previous crash," "prior crash," and "a prior" that had probability scores greater than 0.846. Only the unigram "previous" with probability score 0.867

at the top of the list could be related to the secondary crash. It suggests that the probability scores of the words greater than or equal to 0.846 is an appropriate cutoff value for identifying secondary crashes from narratives. Due to the small training dataset, it is possible that crucial information regarding secondary crashes is missing; as a result, this dataset cannot be used to draw a definitive conclusion. In addition, the best results were found with cut-off values of 0.7 for word probability and 0.8 for classification score (APPENDIX G). Thus, a word probability of 0.7 was used as the cut-off value for test dataset. The tool took approximately 10 seconds to upload test data and run the model.

Figure 7.5-1 shows that among the 40,516 crashes, 2177 (5.37%) have a classification score that is higher than the selected cut-off value (0.80) and can be considered a possible secondary crash. Among the 2,177 crashes, 287 are flagged as secondary crashes by police officers in the crash data, indicating that the model is well trained. In addition, the tool found another 1,890 probable secondary crash candidates.



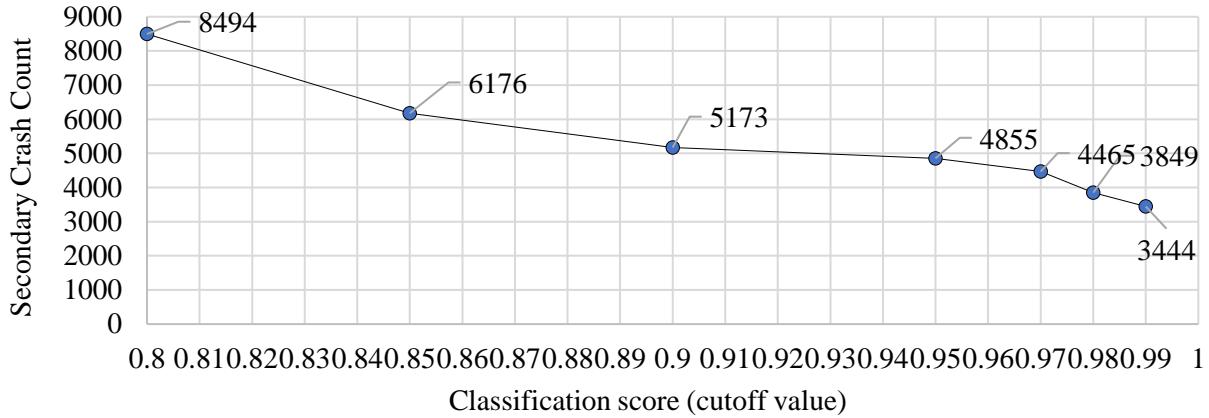
**Figure 7.5-1 Number of Secondary Crashes vs. Cut-off Value of the Classification Score**

## 7.5.2 Case Study with Large Dataset

For the larger dataset, the model was trained on a DELL laptop with an Intel(R) Core (TM) i7-8565U CPU @ 1.80GHz 1.99 GHz processor, 16.00GB of RAM, and 64-bit Windows in approximately 10 seconds. Compared to the previous model, this model discovered more unigrams and bigrams related to secondary crashes (APPENDIX H). The unigrams ‘secondary’ and ‘previous’ with higher probability scores ( $> 0.87$ ) were related to secondary crashes. The top bigrams were ‘a previous’, ‘previous crash’, ‘a secondary’, ‘secondary crash’, and ‘crash ahead’ with probability scores greater than 0.93. It suggests that the classification scores of the narratives and the probability values of the words greater than or equal to 0.95 can be a suitable cutoff value for identifying secondary crashes from narratives using this model. Many other unigrams and bigrams such as ‘median wall’, ‘median shoulder’, and ‘traffic congestion’ in the top 100 list are not related to secondary crashes but have a very high probability score. These words are commonly found in all types of crash narratives and can affect the classification score. The best results were found with a cut-off value of 0.8 for word probability and 0.95 for classification score (APPENDIX I).

The tool took approximately 10 seconds to upload test data and run the model. In Figure 7.5-2, the results show that among 40,516 crashes, 4,855 (11.98%) have a classification score that is higher than the selected cut-off value, so those were not considered to be possible secondary crashes. This model flagged 287 secondary crashes out of 4,855, the same as the previous model. The tool found another 4,568 probable secondary crashes that can be further investigated. As aforementioned, this model discovered more secondary crash-related unigrams

and bigrams, resulting in an increase in the number of probable secondary crashes when compared to the previous model.



**Figure 7.5-2 Secondary Crashes vs. the Cut-off Value of the Classification**

The case study with a larger dataset reveals that the tool discovers many unigrams and bigrams related to secondary crashes, resulting in more positive cases than the case study with a smaller dataset. However, due to the presence of many high frequency common words that are not related to secondary crashes in the training dataset, the test results may produce more false positive cases. NoisyOR performs well when crash narratives contain more high frequency words related to a specific crash type, such as in the case of distracted driving or work zone crash classification. In the narratives of secondary crashes, there were no observations of a narrative containing multiple words related to secondary crashes. In a scenario such as this, the unigram probability and the bigram probability of NoisyOR can provide valuable insight about the narrative and help us determine whether or not to use the NoisyOR classifier. If there are few words with high probability values, only unigrams and bigrams will be able to classify the narratives. If the probability value of significant unigrams and bigrams is low, such as 0.70 or

0.75, and the narrative contains very few positive words, it will be difficult for NoisyOR to retrieve true positives from the narrative.

## 7.6 Conclusions and Recommendations

The purpose of this research is to develop CIEACT, an online web tool capable of analyzing and extracting valuable information from crash narratives using text mining techniques. The tool is based on the Django web-framework that uses pythons for scripting and provides easy integration of HTML and a relational database. CIEACT provides an opportunity to develop multiple models concurrently and analyze the results. CIEACT was tested both offline and online using different types of electronic devices. During the testing process, the tool showed satisfactory performance with all designed functionalities working properly. A case study was also conducted to evaluate the functionality of a custom model that allows users to train their model in real time. The tool shows satisfactory performance in classifying secondary crashes.

CIEACT can assist safety practitioners and professionals in extracting valuable information from narratives, recovering missed crashes, and reviewing the results in both tabular and spatial contexts. Federal and state transportation agencies can use this tool to analyze crashes and implement new policies to enhance structural data quality and determine effective safety measures.

The most attractive aspect of CIEACT is its simplicity and flexibility. The user interface includes detailed instructions, allowing a user to easily navigate to different webpages within the tool. In terms of flexibility, the tool provides both a default built-in model and the ability to train

a model in real time for crash analysis and classification. The tool presents the result in tabular format with color coding and classification score. Finally, users can download the result in a csv file for further review and analysis.

CIEACT gives users the option of training their own models or using a default model that has already been trained. It uses a relational database PostgreSQL to store information from the default model that can be used to update any existing default model or to add a new model. The tool handles the default model at the database level and the user-trained model at the memory level to reduce computational time. The current version of CIEACT uses a NoisyOR based hybrid (unigram + bigram) classifier for both the default model and the training of a new model. The default model is applicable for work zone crash classification. However, the tool has been designed in such a way that its functionality can be scaled up by integrating other text mining algorithms. Future versions of the tool are expected to include additional crash classification models, such as distracted, inattentive, and secondary, as well as different versions of models, such as NoisyOR-unigram, NoisyOR-bigram, and NoisyOR-trigram. Future versions will also include machine-learning algorithms.

## **Chapter 8. CONCLUSION AND FUTURE RECOMMENDATIONS**

The alarming rise in traffic-related deaths and injuries calls for innovative and comprehensive methods to be developed for analyzing what led to a motor vehicle crash and why a crash happened. The primary source for traffic safety analysis is the structure (also called tabular) data collected from crash reports. However, structure data is insufficient because of missing information, incomplete sequence of events, misclassified crash types, and other issues. A safety professional may manually review hundreds of or even thousands of crash narratives, a form of unstructured text data, in order to search the missing information and identify new aspects and circumstances pertaining to a crash. To improve the efficiency and accuracy of highway safety analysis, interest in using NLP and ML techniques and text analytics in crash data analysis has grown rapidly. The new strategies include examining problems from various viewpoints, maximizing the utilization of available data, and using the most appropriate tools and technologies within and outside the traffic safety area.

The primary goal of this dissertation is to identify and develop the models, techniques, and algorithms to improve traffic safety analysis using text analytic methods. The dissertation is composed of seven chapters. A brief summary of each chapter is provided below.

In chapter 3, NoisyOR, a keyword-based text classifier, has been developed to identify misclassified work zone (WZ), distracted (DD) and Inattentive (ID) crashes from the crash narratives. NoisyOR does not have data imbalance issues, requires little training time, and is computationally efficient and easier to implement. Therefore, this method is highly recommended to real-world applications. The developed NoisyOR algorithm is capable of

identifying missed crashes from noisy narratives in an automatic and interpretable way. The findings of this research underscore the importance of properly documenting crash information in the crash narrative.

In chapter 4, other classification methods such as multinomial naive bayes (MNB), logistic regression (LGR), support vector machine (SVM), k-nearest neighbor (K-NN), random forest (RF) and gated recurrent unit (GRU) have been developed for classifying crashes from narrative data. The comparison of the model results indicates that GRU and NoisyOR perform the best, followed by MNB, RF, and afterward K-NN. As top performers, GRU is complex, computationally intensive and difficult to interpret; while NoisyOR is simple, computationally efficient, and theoretically sound. Nevertheless, the results from both methods complement each other and therefore, combining both methods may help determine the maximum number of missed crashes. In addition, several narrative-related issues are identified and solutions to help model selection are proposed when working with crash narratives.

In chapter 5, a novel neural network architecture named Sentence-based Hierarchical Attention Network (SHAN) has been developed to classify crashes. The model used word and sentence level attention layer with the gated recurrent unit (GRU) neural network to generate attention score for each sentence and word. Then, the sentence with the highest attention score was used for classification. The results of SHAN outperformed GRU and Hierarchical Attention Network (HAN). During the model training process, it was found that imbalanced and noisy text data drastically degrades the performance of ML models. To address data imbalance issues, the undersampling strategy was applied towards the negative cases. The Latent Dirichlet allocation (LDA) method was utilized to group similar narratives, and then random samples were drawn

from each group to balance the data. Overall, SHAN handled noisy or irrelevant parts of narratives effectively, including narratives of varying lengths and the model is effective and stable in classifying crashes. This is a significant improvement over the previous ML model.

In chapter 6, several crash sequence generation algorithms such as the Part-of-Speech tagging (PT), Pattern Matching with POS Tagging (PMPT), Dependency Parser (DP), and Hybrid Generalized (HGEN) algorithms have been developed and tested thoroughly using crash narratives. During the development of the algorithms, several ML and NLP algorithms and tools were tested, and the most effective one was chosen. The best results were achieved with the Hybrid Generalized (HGEN) algorithm. HGEN uses a predefined events and event-related action words from single vehicle crash narratives and Python package “Spacy” for NLP pipeline and text mining. The algorithm found many missing and additional events which can complement the relevant information stored in the structure data. Besides, the association analysis simplifies the complex information about how events are related to each other. When designing a safety policy or implementing any safety measure, a safety practitioner can use such information to select the common subsequent of events that occurs prior to the most harmful event and to gain a better understanding of the crash generating process.

In chapter 7, the crash information extraction, analysis and classification tool (CIEACT), a simple and flexible online web tool has been developed and implemented to analyze crash narratives using text mining techniques. The tool uses a python-based Django web-framework, HTML and a relational database (PostgreSQL). CIEACT enables concurrent model development and analysis. It has a default built-in model to wok zone related crashes. It can also develop a model in real time given the training data. The device compatibility of the tool has been tested

offline and online on a variety of devices. The user interface provides detailed instructions for navigating the tool. The tool displays narratives by color-coding the most important words and classification score in a tabular format, as well as displaying crashes on an embedded map. The user can save the results in a csv file, along with the words and the probability scores.

The most important contribution of this research is to demonstrate the significance of text analytics in traffic safety from multiple perspectives, including:

- 1) The literature review in this dissertation concludes that ML techniques work differently to solve the crash classification problem, making methodology selection difficult. None of the previous studies discussed the model performance results in light of narrative characteristics. This study investigated narratives in depth and identified narrative-related issues that can be used to guide the model selection.
- 2) Safety practitioners rely on manual review to recover crashes from crash narratives, which is time consuming and labor intensive. The NoisyOR classifier developed from this dissertation can identify missed crashes from noisy narratives in an automatic and interpretable way. Furthermore, NoisyOR does not have any data imbalance issues, requires little training time, is computationally efficient and easier to implement. Therefore, this method will be applicable to real-world problems.
- 3) No previous transportation study used attention layers with deep neural network (DNN) to classify crashes using narratives. As a result, the value of attention layers over traditional DNN has not been assessed. The proposed SHAN model is effective in classifying crashes and interpreting results by demonstrating attention scores for sentences and words. SHAN handled noisy or irrelevant parts of narratives

effectively, including narratives of varying lengths. The visualization of the result by attention weight allows for a faster review. Furthermore, the LDA model was used for data balancing during the undersampling process. When computational power and data storage capacity are limited, undersampling is an excellent method for dealing with large and unbalanced datasets.

- 4) Compared to structure data, crash sequence generated by the HGEN algorithm offers new and additional insights of crash occurrence. The association analysis simplifies the complex interrelationship between events. When designing a safety policy or implementing any safety measure, safety practitioners can be benefited by this information. They can select the most frequent subsequence of events that occur before the most harmful event and entangle the complexity of crash propagation.
- 5) A web-based tool CIEACT can help safety practitioners and professionals to automate crash information extraction, recover missed crashes or classify a new crash from narratives. Users can also train their own models or utilize a predefined model in CIEACT and review the results in tabular and spatial context.

Based on the demonstration and discussion of the importance and opportunities of text analytics in traffic safety analysis, the future research directions are outlined as follows:

- 1) As shown in the dissertation, the effectiveness of NoisyOR and GRU in recovering missed crashes from narratives is complementary. Therefore, new research is needed to explore the use of a mixture of techniques, or cocktail treatment.
- 2) Some advance pretrained model, such as XLNet (Zhilin Yang et al., 2019), ERNIE(S. Wang et al., 2021), Transformer (Vaswani et al., 2017), and BERT (Devlin et al., 2018),

have showed promising result in text classification. These models can be leveraged with SHAN in future research.

- 3) The findings of crash sequence generation show that some events, such as cross centerline and run off roadway, are not mentioned explicitly in the narratives. Moreover, some narratives are noisy because of repeated information from the driver, witnesses, and police officers. Due to the complexity of the scenarios, additional research is necessary to make the HGEN algorithm more robust.
- 4) CIEACT can be expanded by adding other text mining techniques. The next version of the tool will incorporate distracted, inattentive, and secondary crash categorization models as well as NoisyOR-unigram, NoisyOR-bigram, and NoisyOR-trigram models. Another area for potential future work is the integration of ML algorithms with CIEACT.
- 5) In addition to crash narratives, other textual data such as social media, medical records, and insurance claims, can be investigated to enhance traffic safety analysis.

## REFERENCES

- Abay, K. A. (2015). Investigating the nature and impact of reporting bias in road crash data. *Transportation Research Part A: Policy and Practice*, 71, 31–45.  
<https://doi.org/10.1016/j.tra.2014.11.002>
- Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *ArXiv Preprint ArXiv:1707.02919*.
- Amoros, E., Martin, J. L., & Laumon, B. (2006). Under-reporting of road crash casualties in France. *Accident Analysis and Prevention*, 38(4), 627–635.  
<https://doi.org/10.1016/j.aap.2005.11.006>
- Arteaga, C., Paz, A., & Park, J. W. (2020). Injury severity on traffic crashes: A text mining with an interpretable machine-learning approach. *Safety Science*, 132(May), 104988.  
<https://doi.org/10.1016/j.ssci.2020.104988>
- Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *ArXiv Preprint ArXiv:1409.0473*.
- Bauder, R. A., & Khoshgoftaar, T. M. (2018). The effects of varying class distribution on learner behavior for medicare fraud detection with imbalanced big data. *Health Information Science and Systems*, 6(1), 9. <https://doi.org/10.1007/s13755-018-0051-3>
- Blackman, R., Debnath, A. K., & Haworth, N. (2020). Understanding vehicle crashes in work zones: Analysis of workplace health and safety data as an alternative to police-reported

crash data in Queensland, Australia. *Traffic Injury Prevention*, 21(3), 222–227.

<https://doi.org/10.1080/15389588.2020.1734190>

Boser, B. E., Vapnik, V. N., & Guyon, I. M. (1992). Training Algorithm Margin for Optimal Classifiers. *Perception*, 144–152.

Breiman, Leo. (1999). Randon Forests. *Machinelearning202.Pbworks.Com*, 1–35.

[http://machinelearning202.pbworks.com/w/file/fetch/60606349/breiman\\_randomforests.pdf](http://machinelearning202.pbworks.com/w/file/fetch/60606349/breiman_randomforests.pdf)

Breiman, LEO. (2001). Random forests. *Random Forests*, 1–122.

<https://doi.org/10.1201/9780367816377-11>

Brindha, S., Prabha, K., & Sukumaran, S. (2016). A survey on classification techniques for text mining. *ICACCS 2016 - 3rd International Conference on Advanced Computing and Communication Systems: Bringing to the Table, Futuristic Technologies from Around the Globe*, 01(i), 1–5. <https://doi.org/10.1109/ICACCS.2016.7586371>

Brown, D. E. (2016). Text Mining the Contributors to Rail Accidents. *IEEE Transactions on Intelligent Transportation Systems*, 17(2), 346–355.

<https://doi.org/10.1109/TITS.2015.2472580>

Campbell, J. C., Hindle, A., & Stroulia, E. (2003). Latent Dirichlet Allocation: Extracting Topics from Software Engineering Data. *The Art and Science of Analyzing Software Data*, 3, 139–159. <https://doi.org/10.1016/B978-0-12-411519-4.00006-9>

CDC, C. for D. C. and P. (2020). *No Title*.

<https://www.cdc.gov/niosh/topics/highwayworkzones/default.html>

Chang, Y., & Lin, C. (2008). Feature Ranking Using Linear SVM. *Feature Ranking Using Linear SVM*, 53–64.

Cheng, Y., Parker, S., Ran, B., & Noyce, D. (2012). Enhanced analysis of work zone safety through integration of statewide crash and lane closure system data. *Transportation Research Record*, 2291, 17–25. <https://doi.org/10.3141/2291-03>

Cho, K., van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). *On the Properties of Neural Machine Translation: Encoder–Decoder Approaches*. 103–111.  
<https://doi.org/10.3115/v1/w14-4012>

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.  
<https://doi.org/10.1007/BF00994018>

Cuingnet, R., Rosso, C., Chupin, M., Lehéricy, S., Dormont, D., Benali, H., Samson, Y., & Colliot, O. (2011). Spatial regularization of SVM for the detection of diffusion alterations associated with stroke outcome. *Medical Image Analysis*, 15(5), 729–737.  
<https://doi.org/10.1016/j.media.2011.05.007>

Cunningham, P., & Delany, S. J. (2020). k-Nearest neighbour classifiers 2nd edition (with python examples). *ArXiv*, 1, 1–22.

Daniel, J., Dixon, K., & Jared, D. (2000). Analysis of fatal crashes in Georgia work zone. *Transportation Research Record*, 1715, 18–23. <https://doi.org/10.3141/1715-03>

Das, S., Datta, S., Zubaidi, H. A., & Obaid, I. A. (2021). Applying interpretable machine learning to classify tree and utility pole related crash injury types. *IATSS Research*, 45(3),

310–316. <https://doi.org/10.1016/j.iatssr.2021.01.001>

Das, S., Dutta, A., & Tsapakis, I. (2021). Topic models from crash narrative reports of motorcycle crash causation study. *Transportation Research Record*, 2675(9), 449–462.  
<https://doi.org/10.1177/03611981211002523>

Das, S., Le, M., & Dai, B. (2020). Application of machine learning tools in classifying pedestrian crash types: A case study. *Transportation Safety and Environment*, 2(2), 106–119.  
<https://doi.org/10.1093/tse/tdaa010>

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv Preprint ArXiv:1810.04805*.

Dong, Y., Liu, P., Zhu, Z., Wang, Q., & Zhang, Q. (2020). A Fusion Model-Based Label Embedding and Self-Interaction Attention for Text Classification. *IEEE Access*, 8, 30548–30559. <https://doi.org/10.1109/ACCESS.2019.2954985>

Du, J., Gui, L., He, Y., Xu, R., & Wang, X. (2019). Convolution-based neural attention with applications to sentiment classification. *IEEE Access*, 7, 27983–27992.  
<https://doi.org/10.1109/ACCESS.2019.2900335>

Elias, A. M., & Herbsman, Z. J. (2000). Risk analysis techniques for safety evaluation of highway work zones. *Transportation Research Record*, 1715, 10–17.  
<https://doi.org/10.3141/1715-02>

Elvik, R., & Mysen, A. (1999). Incomplete accident reporting: meta-analysis of studies made in 13 countries. *Transportation Research Record*, 1665(1), 133–140.

Farmer, C. M. (2003). Reliability of police-reported information for determining crash and injury severity. *Traffic Injury Prevention*, 4(1), 38–44. <https://doi.org/10.1080/15389580309855>

Feng, A., Zhang, X., & Song, X. (2022). Unrestricted Attention May Not Be All You Need—Masked Attention Mechanism Focuses Better on Relevant Parts in Aspect-Based Sentiment Analysis. *IEEE Access*, 10, 8518–8528. <https://doi.org/10.1109/ACCESS.2022.3142178>

Fitzpatrick, C. D., Rakasi, S., & Knodler, M. A. (2017). An investigation of the speeding-related crash designation through crash narrative reviews sampled via logistic regression. *Accident Analysis and Prevention*, 98, 57–63. <https://doi.org/10.1016/j.aap.2016.09.017>

Gao, L. (2022). *Verb-Based Text Mining of Road Crash Report. July 2012.*

Gao, L., & Wu, H. (2013). Verb-Based Text Mining of Road Crash Report. *Transportation Research Board, 92nd Annual Meeting*, 5–16. <http://trid.trb.org/view/2013/C/1241434>

Garber, N. J., & Zhao, M. (2002). Distribution and characteristics of crashes at different work zone locations in Virginia. *Transportation Research Record*, 1794, 19–28. <https://doi.org/10.3141/1794-03>

Gers, F. A., & Cummins, F. (1999). *Learning to Forget : Continual Prediction with LSTM*. 470, 850–855.

Glen, S. (2019). *Undersampling and Oversampling in Data Analysis" From StatisticsHowTo.com: Elementary Statistics for the rest of us!* <https://www.statisticshowto.com/undersampling/>

Graham, J. L., & Migletz, J. (1983). Collection of Work-Zone Accident Data. *Transportation Research Record*, 15–18.

Graham, J. L., Paulsen, R. J., & Glennon, J. C. (1978). Accident analysis of highway construction zones. *Transportation Research Record*, 693, 25–32.

Gupta, V., Lehal, G. S., & others. (2009). A survey of text mining techniques and applications. *Journal of Emerging Technologies in Web Intelligence*, 1(1), 60–76.

Guyon, I., Weston, J., Barnhill, S., Labs, T., & Bank, R. (2013). Tracking cellulase behaviors. *Biotechnology and Bioengineering*, 110(1), fmvi-fmvi. <https://doi.org/10.1002/bit.24634>

Ha, S., Marchetto, D., Burke, M. E., & Asensio, O. I. (2020). Detecting behavioral failures in emerging electric vehicle infrastructure using supervised text classification algorithms. *Transportation Research Board Annual Meeting*, 20.

Hao, S., Lee, D. H., & Zhao, D. (2019). Sequence to sequence learning with attention mechanism for short-term passenger flow prediction in large-scale metro system. *Transportation Research Part C: Emerging Technologies*, 107(July), 287–300. <https://doi.org/10.1016/j.trc.2019.08.005>

Hauer, E., & Hakkert, A. S. (1988). Extent and some implications of incomplete accident reporting. *Transportation Research Record*, 1185(1–10), 17.

Hauer, Ezra, & Hakkert, A. S. (1988). Extent and some implications of incomplete accident reporting. *Transportation Research Record*, 1185(January), 1–10.

Hearst, M. (2003). *What Is Text Mining?* <https://people.ischool.berkeley.edu/~hearst/text-mining.html>

Heidarysafa, M., Kowsari, K., Barnes, L., & Brown, D. (2019). Analysis of Railway Accidents' Narratives Using Deep Learning. *Proceedings - 17th IEEE International Conference on Machine Learning and Applications, ICMLA 2018*, 1446–1453.  
<https://doi.org/10.1109/ICMLA.2018.00235>

Ho, T. K. (1995). Random decision forests. *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, 1, 278–282.  
<https://doi.org/10.1109/ICDAR.1995.598994>

Hoffman, M. D., Blei, D. M., & Bach, F. (2010). Online Learning for latent Dirichlet allocation (Supplementary Material). *Nature*, 1–9. <http://papers.nips.cc/paper/3902-online-learning-for-latent-dirichlet-allocation.pdf>

Inzalkar, S., & Sharma, J. (2015). A survey on text mining-techniques and application. *International Journal of Research In Science & Engineering*, 24, 1–14.

Japkowicz, N. (2000). The class imbalance problem: Significance and strategies. *Proc. of the Int'l Conf. on Artificial Intelligence*, 56, 111–117.

Jeong, H., Jang, Y., Bowman, P. J., & Masoud, N. (2018). Classification of motor vehicle crash injury severity: A hybrid approach for imbalanced data. *Accident Analysis and Prevention*, 120(December 2017), 250–261. <https://doi.org/10.1016/j.aap.2018.08.025>

Kantardzic, M. (2011). *Data mining: concepts, models, methods, and algorithms*. John Wiley &

Sons.

Kenton, M. C., Kristina, L., & Devlin, J. (1953). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Mlm.*

Khattak, A. J., Khattak, A. J., & Council, F. M. (2002). Effects of work zone presence on injury and non-injury crashes. *Accident Analysis and Prevention*, 34(1), 19–29.  
[https://doi.org/10.1016/S0001-4575\(00\)00099-3](https://doi.org/10.1016/S0001-4575(00)00099-3)

Korde, V., & Mahender, C. N. (2012). Text classification and classifiers: A survey. *International Journal of Artificial Intelligence & Applications*, 3(2), 85.

Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information (Switzerland)*, 10(4), 1–68.  
<https://doi.org/10.3390/info10040150>

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4), 221–232. <https://doi.org/10.1007/s13748-016-0094-0>

Kwayu, K. M., Kwigizile, V., Lee, K., & Oh, J. S. (2021). Discovering latent themes in traffic fatal crash narratives using text mining analytics and network topology. *Accident Analysis and Prevention*, 150(November 2020), 105899. <https://doi.org/10.1016/j.aap.2020.105899>

Leevy, J. L., Khoshgoftaar, T. M., Bauder, R. A., & Seliya, N. (2018). A survey on addressing high-class imbalance in big data. *Journal of Big Data*, 5(1), 42.  
<https://doi.org/10.1186/s40537-018-0151-6>

Leximancer. (2010). *Leximancer white paper*. 7.

[https://www.leximancer.com/lmedia/Leximancer\\_White\\_Paper\\_2010.pdf](https://www.leximancer.com/lmedia/Leximancer_White_Paper_2010.pdf)

Li, C., Bai, L., Liu, W., Yao, L., & Waller, S. T. (2021). A multi-task memory network with knowledge adaptation for multimodal demand forecasting. *Transportation Research Part C: Emerging Technologies*, 131(August), 103352. <https://doi.org/10.1016/j.trc.2021.103352>

Li, Y., & Bai, Y. (2009a). Effectiveness of temporary traffic control measures in highway work zones. *Safety Science*, 47(3), 453–458. <https://doi.org/10.1016/j.ssci.2008.06.006>

Li, Y., & Bai, Y. (2009b). Highway work zone risk factors and their impact on crash severity. *Journal of Transportation Engineering*, 135(10), 694–701.

[https://doi.org/10.1061/\(ASCE\)TE.1943-5436.0000055](https://doi.org/10.1061/(ASCE)TE.1943-5436.0000055)

Liddy, E. D. (2001). Natural Language Processing. In Encyclopedia of Library and Information Science. *Marcel Decker, Inc.*, 1–15.

Lin, Y., Shen, S., Liu, Z., Luan, H., & Sun, M. (2016). Neural relation extraction with selective attention over instances. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, 4, 2124–2133. <https://doi.org/10.18653/v1/p16-1200>

Liu, C., & Subramanian, R. (2009). Factors Related to Fatal Single-Vehicle Run-Off-Road Crashes. *URC Enterprises, Inc. National Highway Traffic Administration, November*, 30.

Liu, Z., Lu, C., Huang, H., Lyu, S., & Tao, Z. (2020). Hierarchical /Multi-Granularity Attention-Based Hybrid Neural Network for Text Classification. *IEEE Access*, 8, 149362–149371.

<https://doi.org/10.1109/ACCESS.2020.3016727>

Maghrebi, M., Abbasi, A., Rashidi, T. H., & Waller, S. T. (2015). Complementing Travel Diary Surveys with Twitter Data: Application of Text Mining Techniques on Activity Location, Type and Time. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2015-Octob*, 208–213. <https://doi.org/10.1109/ITSC.2015.43>

Maheswari, M. U., & Sathiaseelan, D. J. G. R. (2017). Text mining: Survey on techniques and applications. *Int. J. Sci. Res.*, 6(6), 45–56.

Manning, C. D., Schütze, H., & Raghavan, P. (2008). *Introduction to information retrieval*. Cambridge university press.

Maria Kränkel, H.-E. L. (2019). *Text Classification with Hierarchical Attention Networks*.

[https://humboldt-wi.github.io/blog/research/information\\_systems\\_1819/group5\\_han/](https://humboldt-wi.github.io/blog/research/information_systems_1819/group5_han/)

Maze, T., Burchett, G., & Hochstein, J. (2005). Synthesis of procedures to forecast and monitor work zone safety and mobility impacts. *Report*.

[http://www.intrans.iastate.edu/publications/\\_documents/t2summaries/wz\\_road\\_close.pdf](http://www.intrans.iastate.edu/publications/_documents/t2summaries/wz_road_close.pdf)

McAdams, R. J., Swidarski, K., Clark, R. M., Roberts, K. J., Yang, J., & Mckenzie, L. B. (2018). Bicycle-related injuries among children treated in US emergency departments, 2006-2015. *Accident Analysis and Prevention*, 118(April), 11–17.

<https://doi.org/10.1016/j.aap.2018.05.019>

Meng, Q., Weng, J., & Qu, X. (2010). A probabilistic quantitative risk assessment model for the long-term work zone crashes. *Accident Analysis and Prevention*, 42(6), 1866–1877.

<https://doi.org/10.1016/j.aap.2010.05.007>

MMUCC. (2012). MMUCC Guideline. In *Nhtsa*.

Nayak, R., Piyatrapoomi, N., & Weligamage, J. (2009). Application of text mining in analysing road crashes for road asset management. *Engineering Asset Lifecycle Management - Proceedings of the 4th World Congress on Engineering Asset Management, WCEAM 2009, September*, 49–58. [https://doi.org/10.1007/978-0-85729-320-6\\_7](https://doi.org/10.1007/978-0-85729-320-6_7)

NHTSA. (2020). *NHTSA 2020 Report*.

<https://one.nhtsa.gov/nhtsa/whatis/planning/2020Report/2020report.html>

Oliveira-Neto, F. M., Han, L. D., & Jeong, M. K. (2009). Tracking large trucks in real time with license plate recognition and text-mining techniques. *Transportation Research Record*, 2121(1), 121–127.

Oniśko, A., Druzdzel, M. J., & Wasyluk, H. (2001). Learning Bayesian network parameters from small data sets: Application of Noisy-OR gates. *International Journal of Approximate Reasoning*, 27(2), 165–182. [https://doi.org/10.1016/S0888-613X\(01\)00039-1](https://doi.org/10.1016/S0888-613X(01)00039-1)

Park, S. H., Synn, J., Kwon, O. H., & Sung, Y. (2018). Apriori-based text mining method for the advancement of the transportation management plan in expressway work zones. *Journal of Supercomputing*, 74(3), 1283–1298. <https://doi.org/10.1007/s11227-017-2142-3>

Proof, J. I. M. P. D. F. (2019). *Application of Machine Learning Tools in Classifying Pedestrian Crash Types*. 1–18.

Qi, B., Costin, A., & Jia, M. (2020). A framework with efficient extraction and analysis of Twitter data for evaluating public opinions on transportation services. *Travel Behaviour and Society*, 21, 10–23.

Rahman, M. M., Strawderman, L., Garrison, T., Eakin, D., & Williams, C. C. (2017). Work zone sign design for increased driver compliance and worker safety. *Accident Analysis and Prevention*, 106(May), 67–75. <https://doi.org/10.1016/j.aap.2017.05.023>

Rakotonirainy, A., Chen, S., Scott-Parker, B., Loke, S. W., & Krishnaswamy, S. (2015). A Novel Approach to Assessing Road-Curve Crash Severity. *Journal of Transportation Safety and Security*, 7(4), 358–375. <https://doi.org/10.1080/19439962.2014.959585>

Roberts, S. C., & Lee, J. D. (2014). *Deciphering 140 Characters : Text Mining Tweets On #DriverDistraction*. 2195–2199.

Sayed, M. A., Qin, X., Kate, R. J., Anisuzzaman, D. M., & Yu, Z. (2021). Identification and analysis of misclassified work-zone crashes using text mining techniques. *Accident Analysis & Prevention*, 159, 106211. <https://doi.org/https://doi.org/10.1016/j.aap.2021.106211>

Serna, A., & Gasparovic, S. (2018). Transport analysis approach based on big data and text mining analysis from social media. *Transportation Research Procedia*, 33, 291–298.

Song, Y., Chitturi, M. V., & Noyce, D. A. (2021). Automated vehicle crash sequences: Patterns and potential uses in safety testing. *Accident Analysis and Prevention*, 153(February), 106017. <https://doi.org/10.1016/j.aap.2021.106017>

Sorock, G. S., Ranney, T. A., & Lehto, M. R. (1996). Motor vehicle crashes in roadway

- construction workzones: An analysis using narrative text from insurance claims. *Accident Analysis and Prevention*, 28(1), 131–138. [https://doi.org/10.1016/0001-4575\(95\)00055-0](https://doi.org/10.1016/0001-4575(95)00055-0)
- Sun, J., & Kim, J. (2021). Joint prediction of next location and travel time from urban vehicle trajectories using long short-term memory neural networks. *Transportation Research Part C: Emerging Technologies*, 128(May), 103114. <https://doi.org/10.1016/j.trc.2021.103114>
- Sun, K., Li, Y., Deng, D., & Li, Y. (2019). Multi-Channel CNN Based Inner-Attention for Compound Sentence Relation Classification. *IEEE Access*, 7, 141801–141809. <https://doi.org/10.1109/ACCESS.2019.2943545>
- Trueblood, A. B., Pant, A., Kim, J., Kum, H. C., Perez, M., Das, S., & Shipp, E. M. (2019). A semi-automated tool for identifying agricultural roadway crashes in crash narratives. *Traffic Injury Prevention*, 20(4), 413–418. <https://doi.org/10.1080/15389588.2019.1599873>
- Tsui, K. L., So, F. L., Sze, N.-N., Wong, S. C., & Leung, T.-F. (2009). Misclassification of injury severity among road casualties in police reports. *Accident Analysis \& Prevention*, 41(1), 84–89.
- Ullman, G. L., Finley, M. D., Bryden, J. E., Srinivasan, R., & Council, F. M. (2008). Traffic Safety Evaluation of Nighttime and Daytime Work Zones. *Transportation Research Board, NCHRP Report 627*. <http://www.trb.org/Publications/Blurbs/160500.aspx>
- Ullman, G. L., & Scriba, T. A. (2004). Revisiting the influence of crash report forms on work zone crash data. *Transportation Research Record*, 1897, 180–182. <https://doi.org/10.3141/1897-23>

- Van Eck, N., & Waltman, L. (2010). Software survey: VOSviewer, a computer program for bibliometric mapping. *Scientometrics*, 84(2), 523–538.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 2017-Decem(Nips)*, 5999–6009.
- Vomlel, J. (2006). Noisy-or classifier. *International Journal of Intelligent Systems*, 21(3), 381–398. <https://doi.org/10.1002/int.20141>
- Wang, J., Hughes, W. E., Council, F. M., & Paniati, J. F. (1996). Investigation of highway work zone crashes: What we know and what we don't know. *Transportation Research Record*, 1529, 54–62. <https://doi.org/10.3141/1529-07>
- Wang, P., Li, J., & Hou, J. (2021). S2SAN: A sentence-to-sentence attention network for sentiment analysis of online reviews. *Decision Support Systems*, 149(September 2020), 113603. <https://doi.org/10.1016/j.dss.2021.113603>
- Wang, S., Sun, Y., Xiang, Y., Wu, Z., Ding, S., Gong, W., Feng, S., Shang, J., Zhao, Y., Pang, C., Liu, J., Chen, X., Lu, Y., Liu, W., Wang, X., Bai, Y., Chen, Q., Zhao, L., Li, S., ...
- Wang, H. (2021). *ERNIE 3.0 Titan: Exploring Larger-scale Knowledge Enhanced Pre-training for Language Understanding and Generation*. <http://arxiv.org/abs/2112.12731>
- Wang, Xinglei, Guan, X., Cao, J., Zhang, N., & Wu, H. (2020). Forecast network-wide traffic states for multiple steps ahead: A deep learning approach considering dynamic non-local spatial correlation and non-stationary temporal dependency. *Transportation Research Part*

*C: Emerging Technologies*, 119(June), 102763. <https://doi.org/10.1016/j.trc.2020.102763>

Wang, Xuesong, Xu, R., Zhang, S., Zhuang, Y., & Wang, Y. (2022). Driver distraction detection based on vehicle dynamics using naturalistic driving data. *Transportation Research Part C: Emerging Technologies*, 136(December 2020), 103561.  
<https://doi.org/10.1016/j.trc.2022.103561>

Weiss, G. M. (2004). Mining with rarity: a unifying framework. *ACM Sigkdd Explorations Newsletter*, 6(1), 7–19.

Weng, J., Zhu, J. Z., Yan, X., & Liu, Z. (2016). Investigation of work zone crash casualty patterns using association rules. *Accident Analysis and Prevention*, 92, 43–52.  
<https://doi.org/10.1016/j.aap.2016.03.017>

WHO. (2022). *Road traffic injuries*. <https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries>

Williams, T., & Betak, J. (2018). A Comparison of LSA and LDA for the Analysis of Railroad Accident Text. *Procedia Computer Science*, 130, 98–102.  
<https://doi.org/10.1016/j.procs.2018.04.017>

Wu, K. F., Sasidharan, L., Thor, C. P., & Chen, S. Y. (2018). Crash sequence based risk matrix for motorcycle crashes. *Accident Analysis and Prevention*, 117(February), 21–31.  
<https://doi.org/10.1016/j.aap.2018.03.022>

Wu, K. F., Thor, C. P., & Ardiansyah, M. N. (2016). Identify sequence of events likely to result in severe crash outcomes. *Accident Analysis and Prevention*, 96, 198–207.

<https://doi.org/10.1016/j.aap.2016.08.009>

Wu, Y., Tan, H., Qin, L., Ran, B., & Jiang, Z. (2018). A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies*, 90(5), 166–180. <https://doi.org/10.1016/j.trc.2018.03.001>

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1–2), 69–90. <https://doi.org/10.1023/A:1009982220290>

Yang, Zhilin, Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32(NeurIPS), 1–18.

Yang, Zichao, Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480–1489.

Yao, R., Zeng, W., Chen, Y., & He, Z. (2021). A deep learning framework for modelling left-turning vehicle behaviour considering diagonal-crossing motorcycle conflicts at mixed-flow intersections. *Transportation Research Part C: Emerging Technologies*, 132(October), 103415. <https://doi.org/10.1016/j.trc.2021.103415>

Ye, F., & Lord, D. (2011a). Investigation of effects of underreporting crash data on three commonly used traffic crash severity models: multinomial logit, ordered probit, and mixed logit. *Transportation Research Record*, 2241(1), 51–58.

Ye, F., & Lord, D. (2011b). Investigation of effects of underreporting crash data on three commonly used traffic crash severity models. *Transportation Research Record*, 2241, 51–58. <https://doi.org/10.3141/2241-06>

Yin, X., Liu, Z., Liu, D., & Ren, X. (2022). A Novel CNN-based Bi-LSTM parallel model with attention mechanism for human activity recognition with noisy data. *Scientific Reports*, 12(1), 1–11. <https://doi.org/10.1038/s41598-022-11880-8>

Zagorecki, A., & Druzdzel, M. (2004). An Empirical Study of Probability Elicitation under Noisy-OR Assumption. *Flairs Conference*, 880–886.

Zhai, G., Yang, Y., Wang, H., & Du, S. (2020). Multi-attention fusion modeling for sentiment analysis of educational big data. *Big Data Mining and Analytics*, 3(4), 311–319. <https://doi.org/10.26599/BDMA.2020.9020024>

Zhang, J., Kwigizile, V., & Oh, J.-S. (2016). Automated hazardous action classification using natural language processing and machine-learning techniques2. In *CICTP 2016* (pp. 1579–1590).

Zhang, K., He, F., Zhang, Z., Lin, X., & Li, M. (2020). Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, 121(February), 102861. <https://doi.org/10.1016/j.trc.2020.102861>

Zhang, X., Green, E., Chen, M., & (Reg) Souleyrette, R. R. (2019). Identifying secondary crashes using text mining techniques. *Journal of Transportation Safety and Security*, 0(0),

1–21. <https://doi.org/10.1080/19439962.2019.1597795>

Zhang, Z., Li, M., Lin, X., Wang, Y., & He, F. (2019). Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies.

*Transportation Research Part C: Emerging Technologies*, 105(October 2018), 297–322.  
<https://doi.org/10.1016/j.trc.2019.05.039>

Zhao, Y., Xu, T., & Hai-feng, W. (2014). Text mining based fault diagnosis of vehicle on-board equipment for high speed railway. *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 900–905.

Zheng, D., Chitturi, M. V., Bill, A. R., & Noyce, D. A. (2015). Analyses of multiyear statewide secondary crash data and automatic crash report reviewing. *Transportation Research Record*, 2514(2514), 117–128. <https://doi.org/10.3141/2514-13>

Zheng, H. T., Wang, W., Chen, W., & Sangaiah, A. K. (2017). Automatic Generation of News Comments Based on Gated Attention Neural Networks. *IEEE Access*, 6, 702–710.  
<https://doi.org/10.1109/ACCESS.2017.2774839>

Zhong, B., Pan, X., Love, P. E. D., Sun, J., & Tao, C. (2020). Hazard analysis: A deep learning and text mining framework for accident prevention. *Advanced Engineering Informatics*, 46(August), 101152. <https://doi.org/10.1016/j.aei.2020.101152>

Zhu, Z., Su, J., & Zhou, Y. (2019). Improving distantly supervised relation classification with attention and semantic weight. *IEEE Access*, 7(Mil), 91160–91168.  
<https://doi.org/10.1109/ACCESS.2019.2925502>

## APPENDIX A: Pseudocode for SHAN

**Algorithm 1:** SHAN

```
input : Narrative Narr, Attention model HAN
output: Most weighted sentence  $S_{mws}$ , SHANscore
1 begin
2    $s_{mws}$  =Most Weighted Sentences
3    $s_{aws}$  =Attention Weigh of Sentence
4    $Narr_{clean}$  =cleanNarrative(Narr)
5    $n$  =Number of Sentences in Narr
6    $N$  =input Array length in HAN
7   SHANscore = -1
8    $S_{mws}$ =None
9   for  $s$  in  $Narr_{clean}$  do
10    |  $Arr_s$ =Array( $Narr_{clean}$ ,  $N$ )
11    |  $s_{mws}$ =MAX( $s_{aws}$  of  $Arr_s$ ,1)
12    |  $Prob_{score}$ = SHAN( $s_{mws}$ )
13    | if  $Prob_{score} > SHAN_{score}$  then
14    |   |  $SHAN_{score} = Prob_{score}$ 
15    |   |  $S_{mws} = s_{mws}$ 
16    | end
17  end
18 end
```

## APPENDIX B: Event description and MMUCC-based event categorization

Collision with Fixed Object			
EVENT	Description	EVENT	Description
SIN PST	Traffic Signpost	BRRAIL	Bridge Rail
TF SIG	Traffic Signal	CULVRT	Culvert
UT PL	Utility Pole	DITCH	Ditch
LTPOLE	Lum Light Support	CURB	Curb
OT PST	Other Post, Pole or Support	EMBKMT	Embankment
TREE	Tree	FENCE	Fence
MAILBOX	Mailbox	OTH FX	Other Fixed Object
GR FAC	Guardrail Face	CABL B	Cable Barrier
GR END	Guardrail End	CONC B	Concrete Traffic Barrier
BRPAR	Bridge Parapet End	OTHR B	Other Traffic Barrier
BRPIER	Bridge/Pier/Abut	UNKN	Unknown
ATTEN	Impact Attenuator/Crash Cushion		
Collision with Person, Motor Vehicle, or Non-Fixed Object			
EVENT	Description	EVENT	Description
MVIT	Motor Vehicle in Transport	OT NMT	Other Non -Motorist
PKVEH	Parked Motor Vehicle	STRUCK	Struck by Falling, Shifting Cargo or Anything Set in Motion by Motor Vehicle
BIKE	Pedal cycle	WZ EQP	Work Zone/Maintenance Equipment
PED	Pedestrian	BRIDGE	Bridge Overhead Structure
TRAIN	Railway Vehicle (Train, Engine)	ANM NA	Non-Domesticated Animal (Alive)
OT RDY	Motor Vehicle in Transport-Other Roadway	ANM DA	Domesticated Animal
OBNFX	Other Object		
Non-collision Events			
EVENT	Description	EVENT	Description
OVRTRN	Overturn/Rollover	ROR R	Run Off Roadway Right
FIRE	Fire/Explosion	ROR L	Run Off Roadway Left
IMMER	Immersion, Full or Partial	CR MED	Cross Median
JKNIF	Jackknife	CR CL	Cross Centerline
OTH NC	Other Non -Collision	DOWN	Downhill Runaway
CARGO	Cargo/Equipment Loss or Shift	REENTR	Reentering Roadway
FELL	Fell/Jumped from Motor Vehicle	SEP	Separation of Units
THRWN	Thrown or Falling Object		

## APPENDIX C: Pseudocode for PT algorithms

**Algorithm 1:** PT

```

input : Text Narrative  $N_{rr}$ 
output: Event Sequences  $E_{seq}$ 
begin
1   Doc = convertNarrativeToTokenWithPOSTag( $N_{rr}$ )
2    $POS$  = Selected List of POS tags
3    $POS_{action}$  = Selected list of POS tags related to action words
4    $T_{ps}$  = Selected tokens with POS tags present in  $POS$ 
5    $T_{tag}$  = POS tag of token T
6    $STOP_{eng}$  = English Stop words
7    $A_{tag}$  = Tag container1
8    $T_c$  = Tag container2
9   for Index,  $T \in T_{ps}$  do
10    |   if  $T_{tag}$  in  $POS_{action}$  &  $T$  is not in  $STOP_{eng}$  then
11    |   |   if  $A_{tag}$  is not "TO" then
12    |   |   |    $E_{seq} += T$ 
13    |   |   |    $T_c = T_{tag}$ 
14    |   |   else
15    |   |   |    $E_{seq} += T[Index - 2] + to + T$ 
16    |   |   end
17    |   end
18    |   if  $T_c$  is verb &  $T_{tag}$  is noun then
19    |   |    $E_{seq} += T$ 
20    |   |    $T_c = T_{tag}$ 
21    |   end
22    |   if  $T_c$  is verb & ( $T_{tag}$  is adverb |  $T_{tag}$  is conjunction) then
23    |   |    $E_{seq} = E_{seq}[:-2] + T$ 
24    |   |    $A_{tag} = T_{tag}$ 
25    |   end
26    |   if  $T_c$  is conjunction &  $T_{tag}$  is noun then
27    |   |    $E_{seq} += T$ 
28    |   |    $A_{tag} = T_{tag}$ 
29    |   end
30    |    $A_{tag} = T_{tag}$ 
31   end
32 end
33 end

```

## APPENDIX D: Pseudocode for PMPT algorithms

**Algorithm 2:** wordTagDic

```

1 [H]   input : List of Narrative  $L_{N_{rr}}$  , list of Gold label Sequences  $L_{G_{Seq}}$ 
2 output: Nested list of POS Tags  $L_{POS}$ 
3 begin
4   Function wordTag( $N_{rr}$ ,  $G_{seq}$ ):
5      $T_{N_{rr}} = \text{NarrativeTokenWithPOSTag}(N_{rr})$ 
6      $T_{tag} = \text{list of token with POS tagging from } G_{seq}$ 
7      $n = \text{Indexer}$ 
8      $m = \text{Counter}$ 
9     while  $n < \text{length of } T_{tag}$  &  $m < \text{length of } T_{N_{rr}}$  do
10     $c = 0$ ;  $d = 0$ 
11    if  $\text{length of } T_{N_{rr}}[m] == 1$  then
12      if  $T_{N_{rr}}[m][0] == T_{tag}[n][0]$  then
13         $U += T_{tag}[n][1]$ 
14         $c = m + 1$ 
15      end
16    end
17    if  $\text{length of } T_{N_{rr}}[m] == 2$  then
18      if  $T_{N_{rr}}[m][0] == T_{tag}[n][0]$  &  $T_{N_{rr}}[m][1] == T_{tag}[n+1][0]$  then
19         $B += T_{tag}[n][1] + T_{tag}[n+1][1]$ 
20         $c = m + 1$ ;  $d = n + 1$ 
21      end
22    end
23    if  $\text{length of } T_{N_{rr}}[m] == 1$  then
24      if  $T_{N_{rr}}[m][0] == T_{tag}[n][0]$  &  $T_{N_{rr}}[m][1] == T_{tag}[n+1][0]$  &  $T_{N_{rr}}[m][2] == T_{tag}[n+2][0]$ 
25        then
26           $T += T_{tag}[n][1] + T_{tag}[n+1][1] + T_{tag}[n+2][1]$ 
27           $c = m + 2$ ;  $d = n + 2$ 
28      end
29    end
30    if  $\text{length of } T_{N_{rr}}[m] == 1$  then
31      if  $T_{N_{rr}}[m][0] == T_{tag}[n][0]$  &  $T_{N_{rr}}[m][1] == T_{tag}[n+1][0]$  &  $T_{N_{rr}}[m][2] == T_{tag}[n+2][0]$ 
32        &  $T_{N_{rr}}[m][3] == T_{tag}[n+3][0]$  then
33           $Q += T_{tag}[n][1] + T_{tag}[n+1][1] + T_{tag}[n+2][1] + T_{tag}[n+3][1]$ 
34           $c = m + 3$ ;  $d = n + 3$ 
35      end
36    end
37     $n = c + 1$ ;  $m = d$ 
38  end
39   $L_{POS} = U + B + T + Q$ 
40  return  $L_{POS}$ 
41 End Function
42 for  $i \in L_{N_{rr}}$  &  $j \in L_{G_{Seq}}$  do
43    $u, b, t, q = \text{wordTag}(i, j)$ 
44    $L_{POS} += u$ ;  $L_{POS} += b$ ;  $L_{POS} += t$ ;  $L_{POS} += q$ 
45 end
46 end

```

**Algorithm 3:** PMPT

```

1 [H]  input : Narrative  $N_{rr}$  , POS tag dictionary  $POS_{dic}$ 
      output: Crash Sequence  $E_{seq}$ 
2 begin
3    $T_{N_{rr}} = \text{NarrativeToTokenWithPOSTag}(N_{rr})$ 
4    $doc_{tag} = \text{list of token with POS tagging from } T_{N_{rr}}$ 
5    $n = \text{Indexer}$ 
6    $k = \text{Counter}$ 
7    $T_{temp} = \text{temporary token holder}$ 
8   while  $n < \text{length of } doc_{tag}$  do
9      $T = doc_{tag}[0]$ 
10     $T_{temp} = []$ 
11    for  $i \in POS_{dic}$  do
12       $i_{len} = \text{CalculateLength}(i)$ 
13      if  $i_{len} == 1 \& doc_{tag}[n][1] == i[0]$  then
14         $T_{temp} = doc_{tag}[n][0]$ 
15         $k = 1$ 
16      end
17      if  $i_{len} == 2 \& doc_{tag}[n][1] == 2 \& doc_{tag}[n][1] == i[0] \& doc_{tag}[n+1][1] == i[1]$  then
18         $T_{temp} = doc_{tag}[n][0] + doc_{tag}[n+1][0]$ 
19         $k = 2$ 
20      end
21      if  $i_{len} == 3 \& doc_{tag}[n][1] == 3 \& doc_{tag}[n][1] == i[0] \& doc_{tag}[n+1][1] == i[1] \&$ 
          $doc_{tag}[n+2][1] == i[2]$  then
22         $T_{temp} = doc_{tag}[n][0] + doc_{tag}[n+1][0] + doc_{tag}[n+2][0]$ 
23         $k = 3$ 
24      end
25      if  $i_{len} == 4 \& doc_{tag}[n][1] == 4 \& doc_{tag}[n][1] == i[0] \& doc_{tag}[n+1][1] == i[1] \&$ 
          $doc_{tag}[n+2][1] == i[2] \& doc_{tag}[n+3][1] == i[3]$  then
26         $T_{temp} = doc_{tag}[n][0] + doc_{tag}[n+1][0] + doc_{tag}[n+2][0] + doc_{tag}[n+3][0]$ 
27         $k = 4$ 
28      end
29    end
30    if  $k == 0$  then
31       $n += 1$ 
32    else
33       $n += k$ 
34    end
35     $E_{seq} += T_{temp}$ 
36  end
37 end

```

## APPENDIX E: POS tags of PMPT methods

Order	POS	Order	POS
1	('VBN', 'IN')	21	('RB',)
2	('DT', 'NN', 'JJ', 'NN')	22	('NN', 'IN', 'DT', 'NN')
3	('JJ', 'NN')	23	('NN',)
4	('VBG', 'RB', 'IN')	24	('VBG', 'IN')
5	('VBD', 'TO', 'VB', 'IN')	25	('DT', 'NN', 'NN')
6	('VBD', 'TO', 'VB')	26	('VBN', 'JJ', 'IN', 'VBG')
7	('IN', 'DT', 'JJ', 'NN')	27	('NNP', 'NNP', 'NNP')
8	('VBD', 'RP')	28	('JJ', 'NN', 'NN')
9	('NN', 'VBD')	29	('NNS',)
10	('JJ',)	30	('DT', 'NN')
11	('VBG', 'JJ')	31	('VBD', 'RB')
12	('VBD', 'RP', 'V рг')	32	('DT', 'NN', 'VBD', 'IN')
13	('VBD',)	33	('NN', 'CC', 'NN')
14	('IN', 'NN')	34	('VBD', 'NN')
15	('NNP',)	35	('NN', 'CC', 'NN', 'NN')
16	('CD', 'NNS')	36	('VBD', 'IN')
17	('DT', 'JJ', 'NN')	37	('NN', 'NN', 'NN')
18	('V рг',)	38	('VBD', 'TO', 'VB', 'V рг')
19	('NN', 'NN')	39	('VBD', 'NNS').
20	('V рг', 'IN', 'DT', 'NN')		

**Algorithm 5:** crashSeq

```
1 [H]   input : Token passed by DP T , controller n
2   output: Crash Sequence Eseq
3   begin
4     SDPact= Selected list of dependency parser
5     SDPobj = Selected list of dependency parser after SDPact parser
6     Tleft =left children of T
7     Tdp =Dependency parser of T
8     if n==0 & Tdp in SDPact then
9       if Tleft[0] is negative then
10         | Eseq +=Tleft[0]+T
11       else
12         | Eseq +=T
13       end
14     end
15     for i ∈ TRight do
16       if length of ileft>0 & ileft[-1] is subject then
17         | break
18       end
19       if idp in SDPact then
20         | Eseq +=i
21         if length of iright>0 then
22           | [crashSeq(isub,0) for isub in iRight]
23         end
24       end
25       if idp in SDPobj then
26         | Eseq +=i
27       end
28   end
```

## APPENDIX F: Pseudocode for DP method

**Algorithm 4:** DP

```

1 [H]
  input : Text Narrative  $N_{rr}$ 
  output: Event Sequences  $E_{seq}$ 
2 begin
3   | Doc =NarrativeToTokenWithPOSTagDependencyParser( $N_{rr}$ )
4   |  $T_{par}$  = Parser of token T
5   |  $Th_{lc}$  = left children of root of T
6   |  $T_{lc}$  = left children of root  $Th$ 
7   for  $T$  in Doc do
8     | if  $T_{par}$  is subject then
9       |   | if length of children of  $T$ >0 then
10      |   |   | if any child of children is numeric then
11      |   |   |   |  $E_{seq} += T_{text} + child_{text}$ 
12      |   |   end
13      |   | else
14      |   |   |  $E_{seq} += T_{text}$ 
15      |   | end
16      |   | if length of  $T_{lc}$ >0 &  $Th_{lc}[0]$  is negative token then
17      |   |   |  $E_{seq} += Th_{lc}[0]$ 
18      |   |   end
19      |   |  $E_{seq} += Th$ 
20      |   | crashSeq( $Th, 1$ )
21      |   end
22    | end
23 end

```

## APPENDIX G: The list of candidate words and phrases extracted from narratives

EVENT CODE	KEYWORDS
ANM	horse, deer, cow, dog
ATTEN	attenuator
BIKE	bicycle, bicyclist, bike
BRRAIL	bridge rail, rail bridge
BRIDGE	bridge
CABL B	cable barrier
CARGO	trailer, tractor
CONC B	concrete barrier, median wall, distress, barrier wall
CR CL	cross centerline, centerline, center line
CR MED	median cross, cross median
CULVRT	culvert
CURB	curb
DITCH	ditch, ravine
EMBKMT	embankment
FENCE	fence
FIRE	smoke, fire
GR END	guardrail end, end guardrail
GR FAC	guardrail face, face guardrail
HYDRNT	fire hydrant, firehydrant
IMMER	submerge
JKNIF	jackknife, jack knife
LT TRN	turn left, left turn
LTPOLE	light pole, street lamp
MAILBOX	mailbox, mail box
OVRTRN	overturn, rollover, roll, rotate, flip
PED	pedestrian, walking, standing
PKVEH	parked
REENTR	reenter, re enter, came back
RT TRN	turn right, right turn
SIN PST	sign post, post sign, stop sign, limit sign, traffic sign, arrow sign, highway sign, road sign, street sign, median sign, parking sign, zone sign, yield sign, exit sign, right sign, left sign
TF SIG	traffic signal, signal pole, signal light, signal post, traffic light
TRAIN	train
TREE	tree, bush, shrubs, brush
UT PL	utility pole, electric pole, power line, telephone pole, utility box
WZ EQP	Barrel, construction

## APPENDIX H: Unigrams for WZ NoisyOR Classifier

The screenshot shows the pgAdmin 4 interface. The left pane displays a tree view of database objects under the 'Browser' tab, including sequences, tables, constraints, indexes, RLS policies, rules, triggers, and various system and application tables. The 'CIEACTapp\_workzoneunigram' table is selected and highlighted with a blue bar at the bottom of its row in the tree.

The right pane contains a 'Query Editor' window with the following SQL query:

```

1 SELECT * FROM public."CIEACTapp_workzoneunigram"
2 ORDER BY id ASC

```

Below the query editor is a 'Data Output' table showing the results of the query. The table has three columns: 'id' (PK bigint), 'Unigrams' (character varying (30)), and 'Probability' (numeric (6,5)). The data consists of 16 rows, each representing a unigram with its ID, the word itself, and its probability score.

<b>id</b> [PK] bigint	<b>Unigrams</b> character varying (30)	<b>Probability</b> numeric (6,5)
1	flagman	0.96000
2	taper	0.94737
3	barreled	0.93750
4	dividers	0.92857
5	roadwork	0.92308
6	kampo	0.91667
7	unfinished	0.91667
8	flaggers	0.91667
9	kucej	0.90909
10	werych	0.90000
11	mit	0.90000
12	menomin	0.90000
13	marquise	0.90000
14	constructions	0.88889
15	tapered	0.88889
16	sommer	0.88889

## APPENDIX I: Top 20 Unigrams and Bigrams from Model Training

<b>Unigrams</b>	<b>Probability</b>	<b>Bigram</b>	<b>Probability</b>
Previous	0.867	previous crash	0.917
International	0.800	prior crash	0.867
Initial	0.750	a prior	0.846
Ln	0.727	from a	0.824
Separate	0.714	following unit	0.800
Four	0.714	disabled vehicle	0.800
Trailers	0.714	the debris	0.800
Jones	0.700	traffic due	0.800
Crashes	0.700	reported that	0.800
Debris	0.667	crash he	0.778
Large	0.667	that there	0.778
Something	0.667	unit 01	0.778
Less	0.667	stopped traffic	0.778
Reduced	0.667	units came	0.778
Hazard	0.667	contact unit	0.778
Pressed	0.667	traffic crash	0.769
Six	0.667	1 semi	0.750
Temporary	0.667	accident in	0.750
Hauling	0.667	stop on	0.750
Gmc	0.667	the international	0.750

## APPENDIX J: Trained Model Summary

CUT-OFF (W- PROB, CL- SCORE)	F-SCORE	PRECISION	RECALL	ACCURACY
(0.25, 0.5)	0.299	0.176	1	0.177
(0.35, 0.5)	0.306	0.181	1	0.205
(0.45, 0.5)	0.331	0.198	1	0.289
(0.5, 0.5)	0.332	0.199	1	0.295
(0.6, 0.5)	0.453	0.294	0.98	0.584
(0.7, 0.5)	0.712	0.593	0.89	0.874
(0.8, 0.5)	0.626	0.81	0.51	0.893
(0.25, 0.6)	0.3	0.176	1	0.181
(0.35, 0.6)	0.309	0.183	1	0.216
(0.45, 0.6)	0.351	0.213	1	0.353
(0.5, 0.6)	0.368	0.225	1	0.396
(0.6, 0.6)	0.453	0.294	0.98	0.584
(0.7, 0.6)	0.712	0.593	0.89	0.874
(0.8, 0.6)	0.626	0.81	0.51	0.893
(0.25, 0.7)	0.3	0.177	1	0.182
(0.35, 0.7)	0.316	0.188	1	0.24
(0.45, 0.7)	0.357	0.217	1	0.367
(0.5, 0.7)	0.377	0.232	1	0.419
(0.6, 0.7)	0.573	0.405	0.98	0.744
(0.7, 0.7)	0.712	0.593	0.89	0.874
(0.8, 0.7)	0.626	0.81	0.51	0.893
(0.25, 0.8)	0.301	0.177	1	0.186
(0.35, 0.8)	0.322	0.192	1	0.26
(0.45, 0.8)	0.382	0.236	1	0.433
(0.5, 0.8)	0.403	0.253	1	0.481
(0.6, 0.8)	0.603	0.436	0.98	0.774
(0.7, 0.8)	<b>0.8</b>	<b>0.821</b>	<b>0.78</b>	<b>0.932</b>
(0.8, 0.8)	0.626	0.81	0.51	0.893
(0.25, 0.9)	0.303	0.179	1	0.195
(0.35, 0.9)	0.34	0.204	1	0.318
(0.45, 0.9)	0.42	0.266	1	0.516
(0.5, 0.9)	0.442	0.284	1	0.558
(0.6, 0.9)	0.717	0.576	0.95	0.868
(0.7, 0.9)	0.796	0.889	0.72	0.935
(0.8, 0.9)	0.455	0.937	0.3	0.874
(0.25, 0.95)	0.307	0.181	1	0.209
(0.35, 0.95)	0.351	0.213	1	0.351
(0.45, 0.95)	0.447	0.288	1	0.567
(0.5, 0.95)	0.483	0.318	1	0.625
(0.6, 0.95)	0.766	0.667	0.9	0.904
(0.7, 0.95)	0.72	0.922	0.59	0.919

CUT-OFF (W- PROB, CL- SCORE)	F-SCORE	PRECISION	RECALL	ACCURACY
(0.8, 0.95)	0.455	0.937	0.3	0.874

Note: Word Probability (W-Prob) and Classification Score (Cl-Score)

## APPENDIX K: Top 100 Unigrams and Bigrams from Model Training

Unigrams	Probability	Bigrams	Probability
Backup	0.93	a previous	0.98
Secondary	0.92	previous crash	0.97
Closure	0.92	94 in	0.96
Primary	0.91	a secondary	0.95
Cruiser	0.9	median wall	0.95
Hazmat	0.9	on i39	0.94
Mp	0.9	up due	0.94
Previous	0.88	median shoulder	0.94
i41	0.87	secondary crash	0.94
Congested	0.87	crash ahead	0.94
None	0.86	39 90	0.93
Barriers	0.85	another crash	0.93
Propelled	0.85	of i	0.93
Interstate	0.84	on interstate	0.92
i39	0.84	i39 90	0.91
39	0.83	traffic slowed	0.91
Ahead	0.83	the primary	0.91
Congestion	0.82	of me	0.91
Distress	0.82	on i41	0.91
Assisting	0.81	43 in	0.91
Marker	0.81	traffic congestion	0.91
41	0.8	traffic was	0.91
Crashes	0.8	41 at	0.9
i94	0.79	separate crash	0.9
Document	0.79	slowing due	0.9
Unrelated	0.79	lane traffic	0.9
Original	0.79	up traffic	0.9
Government	0.78	lane 3	0.9
Temporary	0.78	5 unit	0.89
Blocking	0.78	i 39	0.89
Dep	0.78	just occurred	0.89
Board	0.76	when traffic	0.89
Hudson	0.76	struck no	0.89
85	0.76	other crash	0.89
i90	0.76	another accident	0.89
u3	0.76	to lane	0.88
Reaction	0.76	41 in	0.88
43	0.76	median distress	0.88
94	0.76	i hit	0.88
Reduced	0.74	traffic began	0.88
Tows	0.74	41 unit	0.88
v3	0.74	property was	0.88
Cement	0.74	lanes 2	0.88
90	0.73	chain reaction	0.88
Median	0.73	at mp	0.88
Barrier	0.72	behind upon	0.88
Everyone	0.72	3 came	0.88
2019	0.72	1 semi	0.88
Winnebago	0.72	mile marker	0.88
Highland	0.72	distress lane	0.88

Unigrams	Probability	Bigrams	Probability
Slowing	0.72	unit 5	0.88
Milepost	0.71	ahead unit	0.88
Slowed	0.7	3 vehicle	0.88
Debris	0.7	the disabled	0.88
Semi	0.69	cable barrier	0.88
Separate	0.69	median ditch	0.88
Mile	0.69	lane 4	0.87
2018	0.68	median barrier	0.87
Evasive	0.68	43 at	0.87
Quick	0.68	emergency vehicles	0.87
Disabled	0.68	41 near	0.87
Initial	0.68	multiple vehicle	0.87
Cable	0.68	occurred northbound	0.87
Units	0.67	sudden stop	0.87
Slow	0.67	near mile	0.86
Closed	0.67	slowed due	0.86
Pushing	0.67	got hit	0.86
Blocked	0.67	with emergency	0.86
Overpass	0.67	semi unit	0.86
3	0.66	i94 at	0.86
5	0.66	crash had	0.86
Wall	0.66	was blocking	0.86
Braked	0.66	that crash	0.86
4	0.65	crash north	0.86
Concrete	0.65	construction zone	0.86
145	0.65	43 southbound	0.86
Remain	0.65	i94 in	0.86
Sq	0.65	go traffic	0.86
Emergency	0.64	up ahead	0.86
Rapidly	0.64	state patrol	0.86
0	0.64	happened in	0.86
Braking	0.64	on i	0.86
Abruptly	0.64	traffic ahead	0.86
Visibility	0.63	right distress	0.85
Construction	0.63	interstate 39	0.85
Knifed	0.62	i tried	0.85
Slammed	0.61	was ahead	0.85
Ryan	0.61	middle lane	0.85
Occurred	0.61	license she	0.85
Final	0.61	interstate unit	0.85
Sure	0.6	witnesses upon	0.85
Perpendicular	0.6	4 then	0.85
Sudden	0.6	via radio	0.85
Minutes	0.6	ahead of	0.85
441	0.6	3 stated	0.84
53	0.6	was backed	0.84
Chain	0.6	lane i	0.84
Hard	0.6	the prior	0.84
Avoid	0.6	car behind	0.84
Pushed	0.59	slow in	0.84

## APPENDIX L: Train Model Statistics

CUT-OFF (WPROB, CL-SCORE)	F-SCORE	PRECISION	RECALL	ACCURACY
(0.35, 0.5)	0.335	0.201	0.999	0.253
(0.45, 0.5)	0.367	0.225	0.997	0.354
(0.5, 0.5)	0.375	0.231	0.996	0.375
(0.6, 0.5)	0.476	0.313	0.992	0.589
(0.7, 0.5)	0.61	0.445	0.972	0.767
(0.8, 0.5)	0.759	0.655	0.902	0.892
(0.9, 0.5)	0.726	0.9	0.607	0.914
(0.95, 0.5)	0.235	0.986	0.133	0.837
(0.35, 0.6)	0.335	0.202	0.998	0.256
(0.45, 0.6)	0.381	0.236	0.996	0.392
(0.5, 0.6)	0.411	0.259	0.994	0.465
(0.6, 0.6)	0.476	0.313	0.992	0.589
(0.7, 0.6)	0.61	0.445	0.972	0.767
(0.8, 0.6)	0.759	0.655	0.902	0.892
(0.9, 0.6)	0.726	0.9	0.607	0.914
(0.95, 0.6)	0.235	0.986	0.133	0.837
(0.35, 0.7)	0.344	0.208	0.997	0.286
(0.45, 0.7)	0.385	0.239	0.996	0.402
(0.5, 0.7)	0.421	0.267	0.994	0.486
(0.6, 0.7)	0.543	0.375	0.986	0.688
(0.7, 0.7)	0.61	0.445	0.972	0.767
(0.8, 0.7)	0.759	0.655	0.902	0.892
(0.9, 0.7)	0.726	0.9	0.607	0.914
(0.95, 0.7)	0.235	0.986	0.133	0.837
(0.35, 0.8)	0.356	0.216	0.997	0.321
(0.45, 0.8)	0.414	0.261	0.995	0.47
(0.5, 0.8)	0.45	0.291	0.994	0.543
(0.6, 0.8)	0.566	0.397	0.982	0.717
(0.7, 0.8)	0.707	0.561	0.954	0.851
(0.8, 0.8)	0.759	0.655	0.902	0.892
(0.9, 0.8)	0.726	0.9	0.607	0.914
(0.95, 0.8)	0.235	0.986	0.133	0.837
(0.35, 0.9)	0.375	0.231	0.995	0.376
(0.45, 0.9)	0.447	0.289	0.992	0.539
(0.5, 0.9)	0.487	0.323	0.989	0.608
(0.6, 0.9)	0.617	0.451	0.974	0.773
(0.7, 0.9)	0.73	0.594	0.948	0.868
(0.8, 0.9)	0.813	0.808	0.818	0.929
(0.9, 0.9)	0.726	0.9	0.607	0.914
(0.95, 0.9)	0.235	0.986	0.133	0.837
(0.35, 0.95)	0.397	0.248	0.994	0.433
(0.45, 0.95)	0.478	0.315	0.989	0.594
(0.5, 0.95)	0.527	0.359	0.986	0.667
(0.6, 0.95)	0.652	0.492	0.965	0.806
(0.7, 0.95)	0.781	0.674	0.929	0.902
(0.8, 0.95)	<b>0.813</b>	<b>0.822</b>	<b>0.804</b>	<b>0.931</b>
(0.9, 0.95)	0.555	0.973	0.388	0.883
(0.95, 0.95)	0.235	0.986	0.133	0.837
(0.35, 0.97)	0.414	0.262	0.993	0.472
(0.45, 0.97)	0.499	0.334	0.987	0.628

CUT-OFF (WPROB, CL-SCORE)	F-SCORE	PRECISION	RECALL	ACCURACY
(0.5, 0.97)	0.554	0.386	0.983	0.703
(0.6, 0.97)	0.697	0.548	0.959	0.843
(0.7, 0.97)	0.801	0.71	0.917	0.914
(0.8, 0.97)	0.814	0.844	0.785	0.932
(0.9, 0.97)	0.547	0.975	0.38	0.882
(0.95, 0.97)	0.186	0.991	0.103	0.831
(0.35, 0.98)	0.424	0.27	0.992	0.494
(0.45, 0.98)	0.519	0.352	0.985	0.657
(0.5, 0.98)	0.576	0.408	0.98	0.729
(0.6, 0.98)	0.715	0.572	0.952	0.857
(0.7, 0.98)	0.809	0.729	0.91	0.919
(0.8, 0.98)	0.816	0.88	0.761	0.936
(0.9, 0.98)	0.529	0.974	0.363	0.878
(0.95, 0.98)	0.129	0.986	0.069	0.825

Note: Probability (W-Prob) and Classification Score (Cl-Score)

## APPENDIX M: Homepage (Top) and About page (Bottom) of CIEACT

The screenshot shows the homepage of the CIEACT tool. At the top, there is a navigation bar with links for "About", "Help", and "Contact". Below the navigation bar is the CIEACT logo, which features a red car icon followed by the text "CIEACT". Underneath the logo, the text "Crash Information Extraction Analysis and Classification Tool" is displayed. In the center of the page, there is a section titled "Select Method For Analysis" with two buttons: "Default Model" and "Train Model".

The screenshot shows the "About" page of the CIEACT tool. At the top, there is a navigation bar with links for "Help" and "Contact". Below the navigation bar is the CIEACT logo. The main content area is titled "What is CIEACT?" and includes a brief description: "An online Crash Information Extraction, Analysis and Classification Tool (CIEACT) to analyze and classify crash narrative using text mining and machine learning techniques." Below this, there is a section titled "How it works?" with a note: "The engine of the tool is the models developed from NoisyOr classifier. A user can use either the default trained model or train own model using NoisyOr classifier to analyze text data." There is also a section titled "What to expect?" with a note: "The tool contains an interactive crash map that can display the results and support safety analysis in a spatial context. The functions and analysis offered by CIEACT can provide safety practitioners and professionals with maximum and quick access to information stored in the texts of crash narrative and substantially reduce crash report review time." At the bottom of the page, there is a section titled "Project Details:" which lists the Project ID (MIL117752), Sponsors (US Dept Of Transportation, Wi Dept Of Transportation), and Research Centers (Institute for Physical Infrastructure and Transportation (IPIT)).

## APPENDIX N: Contact page (Top) and Help page (Bottom) of CIEACT

Home

 CIEACT  
Crash Information Extraction Analysis and Classification Tool

Contact Information:

Xiao (Stan) Qiu, Ph.D., P.E.  
Lawrence E. Sosik '73 Professorship  
Professor, Civil and Environmental Engineering  
Director, Institute for Physical Infrastructure and Transportation (IPIIT)  
Founder and Director, Safe and Smart Traffic Lab  
Tel: 414 229-7999  
Email: [qiu@wisc.edu](mailto:qiu@wisc.edu)

Mailing Address:

Civil and Environmental Engineering  
University of Wisconsin-Milwaukee  
SWQ 4414, P.O. Box 784  
Milwaukee, WI 53201-0784

Home

 CIEACT  
Crash Information Extraction Analysis and Classification Tool

Default Model      Train Model

**How to train model on the go:**

- Upload training data in the form of a csv file
- Choose a model type to train
- Provide an email id and a 3-digit pin to run the models. This is required for the tool to distinguish data from multiple users and store individual classification results in the database

**Train Model Page**

**Upload Crash Data And Train Model**

**Step 1: Upload Your Data to Train Model**

Upload a csv file:  Choose File No file chosen

\*Label  
 \*Narrative  
 \*Latitude  
 \*Longitude

**Step 2: Select Model to Train**

NoisyOR (Unigram+Bigram)