AN ABSTRACT OF THE PROJECT REPORT OF

Rui Qin for the degree of Master of Science in Computer Science presented on December 9, 2015.

Title: Data Classification and Extraction of Weather Reports

Abstract approved: _____

Prasad Tadepalli

Severe weather in the United States causes huge insured losses to crop and property frequently. It creates major impact and elicit diverse response in the weather insurance industry. Events like hail, storm, hurricane etc. are more likely to cause catastrophe losses. So it becomes crucial to collect and analyze these extreme weather information. Thus, weather data collection and analysis can be used to define the impacts and responses to extreme weather and to suggest ways that the atmospheric sciences community could work with and assist the industry. We introduce a data pipeline, which consists of three components: extract weather information from non-structured report data, classify each record into different weather events, and finally store these records into a database for research use. The project mainly focuses on the first two components: text classification and information extraction.

Data Classification and Extraction of Weather Reports

by

Rui Qin

A PROJECT REPORT

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented December 9, 2015
Commencement June 2016

Master of Science project report of Rui Qin presented on December 9, 2015.

APPROVED:

Major Professor, representing Computer Science

Director of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my project report will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my project report to any reader upon request.

Rui Qin, Author

ACKNOWLEDGEMENTS

At first, I would like to express my sincere gratitude to my advisor Prof. Prasad Tadepalli for the continuous support of my graduate study and research, for his patience, motivation, enthusiasm, and immense knowledge.

Secondly, I would like to thank my committee: Prof. Xiaoli Fern and Prof. Martin Erwig, for their time and support.

Thirdly, I would like to thank NACSE research staff: Dr. Christopher Daly and Dylan Keon, for their insightful suggestions and feedback.

Last but not the least, I would like to thank my friends and family, especially my husband Jun, who has been supporting and encouraging me all the time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1: Introduction

This project called "Data Classification and Extraction of Weather Reports", is sponsored by Northwest Alliance for Computational Science & Engineering (NACSE), which specializes in designing and constructing Web-based data systems that make scientific data more accessible to scientists and engineers.

One of NACSE's primary activities is the development of large databases that present different "personalities" for different types of users. Many types of audiences may be interested in the information contained in a given database, but the level of detail they want/need will vary. For this project, the customers are interested in the severe weather in the past few years, especially the loss they caused.

Frequent and extremely damaging severe weather conditions in the United States cause huge insured losses, creating major impacts and eliciting diverse responses in the weather insurance industry [8]. There are two fundamental types of weather insurance: crop related and property related. Various weather conditions create crop and property losses. For example, droughts harm crops but do not cause property loss, whereas floods damage both crops and property if they occur in the growing season. In addition, the same weather condition may have different negative impacts based on the severity. For example, small hailstones damage certain sensitive crops, but hailstones larger than 0.75 inch must fall before serious property loss occurs. Such variations between intensity levels and timing of events that cause damage also apply to winds and other weather conditions [8].

Therefore, collecting and analyzing past extreme weather information becomes very important. It could be used to define the impacts and responses to the extreme weather and to suggest ways that the atmospheric sciences community could work with and assist the industry.

This project describes a weather data pipeline, which consists of three steps: extract weather information from source data, classify each record into different weather events, and finally store them into a database for research use.

There are two kinds of information we need to extract. The first is the basic information, like the date, time, and location of the event. The other is the magnitude of the event, particularly for hail. This piece of information is from the description of the event. For example,

- Hailstone happened at 4:00pm, the largest size is golf-ball size.

So the magnitude for the hail is golf-ball size, which is about 1.75 inches.

Besides that, we also did the classification. There are 22 types of weather, like rain, hail, snow, flooding, thunderstorm etc. We use machine learning and natural language tools to classify each event mainly by its description.

The report is organized as follows.

- The related work is reviewed in Chapter 2. We provide related work for two topics, text classification and information extraction.
- In Chapter 3 we firstly describe two data sources of weather reports. After that, we create a parser to parse the non-structured data. The parser mainly focuses on different attributes of each record. Those attributes will be used for the classification and extraction tasks specified in the Chapter 4.
- In Chapter 4 we provide the detailed information on the methods we use. For the text classification problem, we explore different machine learning methods, such as naive bayes and support vector machine. For the information extraction, there are two solutions we propose, chunking and rule-based.
- The experimental results and discussions are presented in Chapter 5.
- We conclude the project and discuss the future work in Chapter 6.

Chapter 2: Related Work

2.1 Text Classification

The goal of text classification (also known as text categorization) is the classification of documents into a fixed number of predefined categories [12]. It has been successfully applied to a variety of domains, including text retrieval, automated metadata generation, word sense disambiguation, web page categorization and in general any application requiring document organization or selection [19].

The first step in text classification is called feature representation, which is to transform documents into a representation suitable for the learning algorithm and the classification task [12]. Features are usually the frequently occurring words or phrases in the document. This way, a document can be represented as a feature vector of term frequencies. However, since almost every domain has a large number of features (for example an English dictionary), there may exist lots of irrelevant and redundant features. Those irrelevant and redundant features tend to have a negative impact on the classification accuracy. Hence, a major step for text classification, called feature selection, is used to reduce the high dimensionality of the feature set. The basic idea of feature selection is based on a term-goodness criterion threshold to eliminate a certain number of features from the full feature space. There are several commonly used criteria, such as document frequency (DF), information gain (IG), mutual information (MI), a χ^2 statistic (CHI), and term strength (TS) [22].

After feature representation and selection, a classifier can be trained based on different machine learning algorithms.

- **k-Nearest Neighbor (k–NN)** is a famous and simple algorithm, it has been applied to text classification since the early stages of the research [10, 20]. To classify a new document, the algorithm searches for the K nearest neighbors of the document, and chooses the class which has the majority number of votes as its final assignment.

- **Decision tree** rebuilds the manual categorization of training documents by constructing well-defined true/false-queries in the form of a tree structure. In a decision tree struc-

ture, leaves represent the corresponding category of documents and branches represent conjunctions of features that lead to those categories [13, 7].

- **Naive Bayes (NB)** classifiers are commonly studied in machine learning [16]. The basic idea of NB approach is to use the joint probabilities of words and classes to estimate the probabilities of classes given a document. The naive part of this method is the assumption of word independence given the class [21]. The NB classifiers require a document model to estimate the probabilities. Two models are usually used: Bernoulli and Multinomial. By empirically comparing their classification performance on different corpora, Bernoulli performs well with small vocabulary sizes, but Multinomial usually performs even better at larger vocabulary sizes [15].

- **Support Vector Machine (SVM)** is a more powerful method for text classification [12]. For SVM, the basic idea is to construct a hyper-plane that separates positive examples from negative examples. It can work for both linearly separable problems as well as for non-linearly separable problems when used with kernels. One remarkable property of SVM is that the learning can be independent of the dimensionality of the feature space.

There are still other commonly used training algorithms for the classification task, such as Regression based methods, Genetic Algorithms, Neural Networks etc. It has been observed that even for a specific classification method, classification performances of the classifiers based on different training text corpuses are different; and in some cases such differences are quite substantial [14].

2.2 Information Extraction

Information extraction (IE) is the task of automatically extracting structured information from unstructured and/or semi-structured machine-readable documents [2]. The classic IE tasks include:

- **Named Entity Recognition (NER)** seeks to locate and classify elements in the text into pre-defined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. [4]. NER task can additionally include extracting descriptive information from the text about the detected entities through

filling of a small-scale template [18]. For example, in the case of person entity extraction, it may include extracting the title, position, nationality, sex, and other attributes of a specific person.

- **Co-reference Resolution (CO)** is the task of finding all mentions (expressions) that refer to the same entity in a text. Coreference resolution research has been an important and active area in the field of Natural Language Processing since the 1960s [17]. As large corpora became available and machine learning methods became more mature, there has been a lot of research aimed at solving the coreference problem using machine learning techniques.

- **Relation Extraction (RE)** is the task of detecting and classifying predefined relationships between entities identified in text. For example, given sentence *"Steve Jobs works for Apple."*, a relation between a person and an organization can be extracted from the sentence: EmployeeOf (Steve Jobs, Apple) [18].

- **Event Extraction (EE)** refers to the task of identifying events in free text and deriving detailed and structured information about them. Usually, event extraction involves extraction of several entities and relationships between them. For instance, extraction of information on terrorist attacks from the text fragment *"Masked gunmen armed with assault rifles and grenades attacked a wedding party in mainly Kurdish southeast Turkey, killing at least 44 people."* involves identification of perpetrators (*masked gunmen*), victims (*people*), number of killed/injured (*at least 44*), weapons and means used (*rifles and grenades*), and location (*southeast Turkey*) [18].

Currently, almost all artificial intelligent methods and machine learning algorithms are used to achieve high performance in Information Extraction from documents. Among the many techniques used, the most basic techniques are syntactic rules and basic Natural Language Processing (NLP) techniques [3]. With the first technique, some syntactic rules and patterns at the word level (such as regular expressions, token-based rules etc.) are used to extract fine information from text. Another widely used technique is based on NLP. The basis idea of using NLP in IE is analyzing grammatical structure at sentence level and then constructing grammatical rules for some useful information within the sentence [9]. Other advanced methods such as Bayesian model, Hidden Markov Model (HMM), Decision Tree etc. are based on basic technologies mentioned above.

2.2.1 Syntactic Rules

Syntactic rules describe string properties in the lowest syntactic level. The most popular syntactic rule is regular expression. Information which conforms to the rules is extracted by pattern matching.

A regular expression is a sequence of characters that define a search pattern, mainly for use in pattern matching with strings. Regular expressions are constructed analogously to arithmetic expressions, by using various operators to combine smaller expressions. For example, expression such as "[A,P]M [0-9]+:[0-9]+" extract time description like "AM 11:45" from documents.

In terms of input, IE assumes the existence of a set of documents in which each document follows a template, i.e. describes one or more entities or events in a manner that is similar to those in other documents but differing in the details [2].

2.2.2 Natural Language Processing

Natural language processing (NLP) techniques rely on syntactic and semantic knowledge which are often manually encoded for a particular domain. Initially NLP was used for machine translation, speech recognition and knowledge representation. Recently, IE researchers use NLP techniques to pre-process documents and to extract underlying information.

Chapter 3: Data Description and Preprocessing

For this chapter, we describe the data sources used in this project. By analyzing data, we find it is necessary to pre-process to transform unstructured data to structured data, which is easier for text classification and information extraction tasks presented in Chapter 4.

3.1 Data Description

Currently, weather information is reported and collected by a lot of communities. The largest provider of daily precipitation observations in the United States is the Community Collaborative Rain, Hail and Snow Network (CoCoRaHS). CoCoRaHS is a community-based network of volunteers of all ages and backgrounds working together to measure and map precipitation (rain, hail and snow), and provide these data for natural resource, education and research applications [1].

The other notable data collection community is State Climate Office of North Carolina (NCSCO). The reason is that North Carolina has a complex climate due to its three distinct regions: the mountains, the piedmont, and the coastal plain. As we know, climate affects many aspects of our daily lives - agriculture, environment, transportation, tourism, and natural disasters to name a few. The mission of NCSCO is to provide climate related services to the state, local and federal agencies, businesses and the citizens of North Carolina. Besides, NCSCO is actively involved in research that enhances its capabilities to provide public service [6].

This weather information reported from the above two communities are representative of various weather conditions in the United States and is used for different research projects. Hence, it is reasonable for us to use it for this project.

It should be noted that the two datasets have different format. The data collected by CoCoRaHS is structured in a tabular form, while the NCSCO data is not well structured. This way, weather reports collected from CoCoRaHS can be imported into database directly. But for data collected by NCSCO, we need to transform the unstructured format to structured format. Figure 3.1 shows an example of a Local Storm Report (LSR) from NCSCO. The weather reported from the same big area (include several cities) will be put together as a batch. So a NCSCO report

parser is necessary for the text classification and information extraction tasks proposed in the next Chapter.

3.2 NCSCO Report Parser

There are two batches in the example of a Local Storm Report shown in Figure 3.1. The first batch contains two events reported from the office at Grand Rapids, MI (Michigan). Two reported events occur respectively in the county of Mason and Muskegon in the state of Michigan. For the second batch, the only event occurred at the county of Erie in the state of PA is reported from the office at Cleveland, OH.

A batch in each report from NCSCO follows a similar pattern. We use the first batch as an example: there are several pieces of information inside each batch, including the office information (line 1-3), issuing information (line 5-7), column titles for event attributes (line 9-11), and attribute values of each event (line 13-21). The second batch specified between the line 29 and line 44 follows the same pattern.

The goal of NCSCO Report Parser is to transform the unstructured information to a structured weather event which can be stored in the database and be easily used for the text classification task. The proposed parser consists of two steps, batch identification and weather record identification. The batch identification is used to extract each batch in the report issued from NCSCO. Based on the batch output of the batch identification, each weather event is extracted and formatted in a tabular form.

3.2.1 Batch Identification

As we described earlier, each batch in each NCSCO report follows the same pattern, starting with three lines description of the office information, in Figure 3.1. The office information of the first batch (line 1-3) in Figure 3.1 is

```
050
NWUS53 KGRR 010214
LSRGRR
```

while that of the second batch (line 29-31) is

```
053
NWUS51 KCLE 010204
LSRCLE
```

```
050
NWUS53 KGRR 010214
LSRGRR

PRELIMINARY LOCAL STORM REPORT
NATIONAL WEATHER SERVICE GRAND RAPIDS MI
913 PM EST WED DEC 31 2014

...TIME...  ...EVENT...      ...CITY LOCATION...     ...LAT.LON...
...DATE...  ....MAG....      ..COUNTY LOCATION..ST.. ...SOURCE....
            ..REMARKS..

0540 PM     NON-TSTM WND GST 8 NNW LUDINGTON         44.06N 86.51W
12/31/2014  M51 MPH          MASON               MI   MESONET

            MESONET STATION BIG SABLE POINT /BSBM4/

0800 PM     NON-TSTM WND GST 4 NW ROOSEVELT PARK     43.23N 86.34W
12/31/2014  M47 MPH          MUSKEGON            MI   MESONET

            MESONET STATION MUSKEGON GLERL /MKGM4/

&&

$$

RICKEY

053
NWUS51 KCLE 010204
LSRCLE

PRELIMINARY LOCAL STORM REPORT
NATIONAL WEATHER SERVICE CLEVELAND OH
904 PM EST WED DEC 31 2014

..TIME...   ...EVENT...      ...CITY LOCATION...     ...LAT.LON...
..DATE...   ....MAG....      ..COUNTY LOCATION..ST.. ...SOURCE....
            ..REMARKS..

0900 PM     SNOW             EDINBORO                41.87N  80.13W
12/31/2014  M2.0 INCH        ERIE                PA   SNOW SPOTTER

            24 HOUR SNOWFALL.

&&

EVENT NUMBER CLE1402076

$$
```

Figure 3.1: LSR File

Hence, we can identify the beginning of each batch, and the middle part between two beginnings is the content of a specific batch. One corner case is the last batch of a report. There is no batch after this one. For this case, we extract the content, between the beginning of the last batch and the end of the report, as the last batch.

This way, we formulate the batch identification problem as the problem of finding the beginning of each batch. After we analyzed the pattern of the beginning sentences for each batch, we find that it is easy to use regular expression to capture the pattern. Here is the a regular expression below:

$$\backslash d\{3\}\backslash n\backslash w\{4\}\backslash d\{1,2\}\ \ \backslash w\{4\}\ \ \backslash d\{6\}.*\backslash n\backslash w\{6\}$$

This regular expression begins with three digits (\d{3}), then follows a new line (\n). The second line starts with four alphanumeric characters (\w{4}), and then one or two digits (\d{1,2}), and then four alphanumeric characters (\w{4}), and then six digits (\d{6}), and ends with an unknown number of other characters (.*). In the third line, there are six alphanumeric characters. This pattern can perfectly capture two office information, "050 NWUS53 KGRR 010214 LSRGRR" and "053 NWUS51 KCLE 010204 LSRCLE".

3.2.2 Weather Record Identification

After the Batch Identification step, we have a collection of batches. Each batch contains one or more weather records. For example, there is a snow event occurred at 0900 PM on 12/31/2014 at the city of Edinboro in Figure 3.1. We define a complete detailed information of a specific weather event as a weather record. The weather record consists of several properties, including TIME, EVENT, CITY LOCATION, LAT and LON, DATE, MAG, and SOURCE etc. The goal of this step is to identify property values for each record in the assumption that we have the same fields in the report except the REMARK field. Figure 3.1 shows that there are three observations for each weather record. The first observation is that each weather record starts with the time, for example 0900 PM. The second observation is that we find that the attribute values have the same order with the property titles. The last observation is that each property value occurs at some fixed position. For example, the EVENT attribute value starts from the 12th position, and the CITY LOCATION attribute value starts from 29th position. This way, we can get attribute values for each record.

Based on the above observations, the weather record identification works in the following way:

- Find the beginning of a specific weather record based on the time.
- After we locate the sentences for each weather record, we extract the property value based on the observation that same property occurs at some fixed position.

There is always a blank line before the time value, either a blank line after property titles or a blank line after a weather record. For the time value, it consists of several parts, including three or four digits (\d{3,4}), one space, "AM" or "PM", and finally followed with four or five spaces (\s{4,5}). Here is the regular expression below:

$$\backslash n \backslash d\{3,4\}\ (AM|PM)\backslash s\{4,5\}$$

After we identify the time value, we get other attribute values for each record by locating the string at a specific position.

By having property values of each record, we format them into a tabular form with a predefined schema shown in Table 3.1. Each row represents a specific property or attribute, such as Date/Time, City, County, and Event etc. Each column represents a specific weather record. For Figure 3.1, there are three weather records in the table. Compared with the original report, we post-process some attribute values in the following way:

- We transform the time value and date value into standard format. Take time value as an example, "913 PM" is represented as "21:13:00". For the date value, we formalize "DEC 31 2014" into "2014-12-31". Then we concatenate both values together.
- We separate LAT/LON into two different attribute values.

Formatting unstructured weather information into structured weather report makes our following text classification and information extraction tasks easier.

Attribute	**Record 1**	**Record 2**	**Record 3**
Date/Time	2014-12-31 17:40:00	2014-12-31 20:00:00	2014-12-31 21:00:00
Time Zone	EST	EST	EST
City	8 NNW LUDINGTON	4 NW ROOSEVELT PARK	EDINBORO
County	MASON	MUSKEGON	ERIE
State	MI	MI	PA
Latitude	44.06	43.23	41.87
Longitude	-86.51	-86.34	-80.13
Event	NON-TSTM WND GST	NON-TSTM WND GST	SNOW
Magnitude	M51 MPH	M47 MPH	M2.0 INCH
Source	MESONET	MESONET	SNOW SPOTTER
Remarks	MESONET STATION BIG SABLE POINT /BSBM4/	MESONET STATION MUSKEGON GLERL /MKGM4/	24 HOUR SNOWFALL.

Table 3.1: Parsed LSR Records

Chapter 4: Methodology

In this chapter, we propose our approaches to classify weather report and extract magnitude information from weather report. Firstly, for weather report classification, we propose a method based on both rule-based approach and machine learning approach. Secondly, we propose two approaches to extract magnitude information from weather report.

4.1 Weather Report Classification

For this section, we would like to classify weather report, shown in Table 3.1, into different categories, such as hail, snow, flooding etc. The total number of categories is 22. The goal is to assign a unique category to each weather report. Hence, we formulate this problem as multiclass classification problem, and mainly work on the dataset collected from NCSCO. The reason is that CoCoRaHS always has the category information (almost always rain, hail or snow) for each weather report, while NCSCO does not have category information in the raw text, and only has the event attribute. So the goal of this task is to assign category information to NCSCO weather report.

By analyzing weather reports collected from NCSCO before, we find that there is a Many-to-One mapping relation between "Event" attribute value (shown in Table 3.1) and weather report category. For example, event value "DUST DEVIL" and "DUST STORM" belong to "Blowing Dust" category, while events "WILDFIRE" and "HEAVY SMOKE" belong to"Fire" category. Table 4.1 shows all events and their corresponding categories. We call these events as "Unique Event" as they can indicate which category a specific weather report should be assigned to.

By doing some initial experiments, we find that almost 99% of NCSCO weather reports can be assigned correct labels based on Table 4.1 with the following rule-based method. In other words, we can use a rule-based method to achieve a super high accuracy.

- Check whether the record's "Event" attribute value belongs to the Unique Event in Table 4.1, if so, we can find its category based on the Many-to-One mapping relation.
- Check whether there is a unique event which has edit distance less or equal to 2 with our

No	Unique Event	Category
1	ROCK SLIDE	Avalanche
2	BLOWING DUST, DUST DEVIL, DUST STORM, HABOOB	Blowing Dust
3	HIGH ASTR TIDES, HIGH SURF, LOW ASTR TIDES, RIP CURRENTS, WAVE HEIGHT, SEICHE, SNEAKER WAVE	Coastal Hazards
4	EXTR WIND CHILL, EXTREME COLD, FREEZE	Cold
5	DENSE FOG, FOG, FREEZING FO	Dense FogG
6	WILDFIRE, HEAVY SMOKE	Fire
7	COASTAL FLOOD, FLASH FLOOD, FLOOD, STORM SURGE, STANDING WATER, MAJ FLASH FLD	Flooding
8	FUNNEL CLOUD	Funnel Cloud
9	HAIL, MARINE HAIL	Hail
10	EXCESSIVE HEAT	Heat
11	HEAVY RAIN, RAIN, MODERATE RAIN	Heavy Rain
12	DOWNBURST, HIGH SUST WINDS, MARINE TSTM WIND, NON-TSTM WND GST, TSTM WND GST, HIGH WINDS, WIND	High Winds
13	FREEZING DRIZZLE, FREEZING RAIN, ICE ACCUMULATION, ICE STORM, ICE	Ice
14	SINK HOLE	Landslide
15	LIGHTNING, LIGHTNING STRIKE	Lightning
16	HEAVY SLEET, SLEET	Sleet
17	BLOWING SNOW, FLURRIES, HEAVY SNOW, ROAD CLOSURES, SNOW, DAMAGING SNOW, RAIN/SNOW MIXED, SNOW DEPTH, STORM TOTAL SNOW, HEAVY WET SNOW	Snow
18	DEBRIS FLOW, NON-TSTM WND DMG, STORM DAMAGE, TSTM WND DMG, MARINE WAVE DMG, WND DMG, WIND DAMAGE, LAND SLIDE, NON-TSTM DMG GST	Storm Damage
19	TROPICAL STORM, TSTM WND MG, WALL CLOUD, WINTER STORM, NON-TSTM WND	Thunderstorm
20	TORNADO, POSSIBLE TORNADO	Tornado
21	WATER SPOUT	Waterspout
22	VOLCANIC ASHFALL	Volcanic Ash

Table 4.1: Many-to-One Mapping between Unique Events and Weather Report Category

input event. The most common typos are usually associated with white space or hyphen, like “WATERSPOUT”, “NONTSTM WND DMG”.

- Check whether there exists a unique event that is a substring of the input event. For example, the input event is "LAKESHORE FLOOD", and we only have event "FLOOD" belong to category "Flooding" in Table 4.1. Because "LAKESHORE FLOOD" is a subtype of "Flooding", so we say "LAKESHORE FLOOD" also belongs to "Flooding".

Based on the findings, we propose our approach in the following way:

1. Use the above rule-based method to assign a category to weather report based on Table 4.1
2. For those weather reports which cannot be assigned a category based on the first step, we predict a category based on a trained machine learning model.

As we review the related work in Chapter 2, Support Vector Machine (SVM) is a commonly used machine learning method for text classification problems, including binary classification and multiclass classification. Hence, we will use the SVM algorithm to train a model in the second step of our approach.

4.1.1 Support Vector Machine

Support vector machine (SVM) is usually an effective method for learning text classifiers. SVM is initially used to learn a binary classifier. Consider N training samples: $\{x_1, y_1\}, \ldots, \{x_N, y_N\}$, where $x_i \in R^m$ is an m-dimensional feature vector representing the i^{th} training sample, and $y_i \in \{-1, 1\}$ is the class label of x_i. A hyperplane in the feature space can be described as the equation $w^T x + b = 0$, where $w \in R^m$ and b is a scalar. When the training samples are linearly separable, SVM yields the optimal hyper-plane that separates the two classes with no training error, and maximizes the minimum distance from the training samples to the hyper-plane. It is easy to find that the parameter pair (w, b) corresponding to the optimal hyper-plane is the solution to the following optimization problem:

$$\begin{aligned} &\text{minimize: } L(w) = \frac{1}{2}||w||^2 \\ &\text{subject to: } y_i(w^T x_i + b) \geq 1, \quad i = 1, ..., N. \end{aligned} \tag{4.1}$$

For linearly nonseparable cases, there is no such hyper-plane that is able to classify every training sample correctly. However the optimization idea can be generalized by introducing the concept of soft margin. The new optimization problem thus becomes:

$$\begin{aligned} &\text{minimize: } L(w,\xi_i) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to: } y_i(w^T x_i + b) \geq 1 - \xi_i, \quad i = 1, ..., N. \end{aligned} \tag{4.2}$$

where ξ_i are called slack variables that are related to the soft margin, and C is the tuning parameter used to balance the margin and the training error.

SVM can be easily extended to multiclass classification problems. Given N training samples: $\{x_1, y_1\}, \ldots, \{x_N, y_N\}$, where $x_i \in R^m$ is an m-dimensional feature vector and $y_i \in \{1, 2, ..., M\}$ is the corresponding class label. The conventional way to extend it to a multi-class scenario is to decompose the M-class problem into a series of two-class problems.

One-against-all approach is the earliest and one of the most widely used implementation [23]. It constructs M binary SVM classifiers, each of which separates one class from all the rest. The i^{th} SVM classifier is trained with all the training examples of the i^{th} class with positive labels, and all the others with negative labels. Mathematically the i^{th} SVM solves the following problem that yields the i^{th} decision function $f_i(x) = w_i^T x + b_i$:

$$\begin{aligned} &\text{minimize: } L(w,\xi_j^i) = \frac{1}{2}||w_i||^2 + C\sum_{i=1}^{N}\xi_j^i \\ &\text{subject to: } \tilde{y_j}(w_i^T x_j + b_i) \geq 1 - \xi_j^i, \quad j = 1, ..., N, \quad \xi_j^i \geq 0 \end{aligned} \tag{4.3}$$

where $\tilde{y_j} = 1$ if $y_j = i$ and $\tilde{y_j} = -1$ otherwise.

At the testing phase, a sample x is classified as in class $i^\star$ whose $f_{i^\star}$ produces the largest value as defined below.

$$i^* = \arg\max_{i=1,...,M} f_i(x) = \arg\max_{i=1,...,M}(w_i^T x + b_i) \tag{4.4}$$

4.2 Magnitude Information Extraction

In this section, we describe how to extract the magnitude of hail from the CoCoRaHS weather report. The reason is that weather reports from NCSCO always have magnitude attribute information, while those from CoCoRaHS reports sometimes don't. Weather reports from CoCoRaHS are reported by volunteers. Not every event reported by people contains the size information. Usually, people give a textual description of the event. So we mainly extract the magnitude

information from the description.

Generally, the hail size is related to a specific object. Table 4.2 shows object descriptions and their sizes, which are used by CoCoRaHS and National Weather Service.

Size	**Value (inch)**	**Size**	**Value (inch)**
rice	0.1	pea	0.25
raisin	0.375	grape	0.5
moth ball	0.5	marble	0.5
dime	0.7	penny	0.75
nickel	0.875	quarter	1
half dollar	1.25	ping pong	1.5
walnut	1.5	golf ball	1.75
hen egg	2	tennis ball	2.5
baseball	2.75	tea cup	3.0
grapefruit	4.00	softball	4.5

Table 4.2: Object-to-Size conversion

Therefore, our goal is to return the size information, shown in Table 4.2, for a given event description. We propose two methods to extract size information: an NLP method and a rule-based method. For both methods, they use the same pro-processing steps. First, the raw text is split into sentences using a sentence segmenter. Then each sentence is further subdivided into words using a tokenizer. The forms after tokenization are the input for our proposed approaches, which we explain in detail in the following sections.

4.2.1 Chunking with Regular Expression

The NLP method we use for size information extraction is called chunking. Instead of parsing the whole sentence, chunking is partial parsing. Chunking is also called shallow parsing which identifies interesting short phrases (like noun phrases) based on Part-Of-Speech (POS) tag information, and indicates whether a word is a noun, verb, or adjective. Table 4.3 shows some of commonly the used POS tags. So, besides sentence segmentation and tokenization, we do POS tagging on tokens of each sentence.

Our goal in this section is to extract magnitude information of hail events from CoCoRaHS reports. We formulate this task as noun phrase extraction (NP-chunking) with key word “size”, like “pea size” and “marble size”. Our approach is that we use a regular expression to create a set

Tag	Description	Tag	Description
CC	Coordinating conjunction	VB	Verb, base form
CD	Cardinal number	DT	Determiner
IN	Preposition or subordinating conjunction	PRP	Personal pronoun
TO	to	VBD	Verb, past tense
JJ	Adjective	VBN	Verb, past participle
NN	Noun, singular or mass	EX	Existential *there*
RB	Adverb	RP	Particle

Table 4.3: POS Tags

of rules, which will be used to capture chunks containing size information. The rules are defined in terms of tag patterns, which are used to describe sequences of tagged words. A tag pattern is a sequence of POS tags delimited using angle brackets, e.g. <DT><JJ><NN>. Tag patterns are similar to regular expression patterns. Take the sentence in Figure 4.1 as an example.

Figure 4.1: Chunking Example 1

We can define a simple grammar with a single tag pattern rule:

```
grammar = "NP: {<NN><TO><JJ><NN>}"
```

This rule says that an NP chunk should be formed whenever the chunker finds a noun (NN) followed by a to (TO), an adjective (JJ) and then a noun (NN). Using this grammar, we create a chunk parser, and test it on the example sentence. It can find the chunk "pea to marble size".

However, a single pattern is not enough because there are so many different sentence structures. It is necessary to define more rules to capture other sentence structures. Take the sentence in Figure 4.2 as another example.

Figure 4.2: Chunking Example 2

For this sentence, the previous grammar cannot find the noun phrase containing size information. Here is the new grammar we define which can handle this example:

```
grammar = "NP: {<NN><TO><JJ><NN> | <NN><IN><DT><NN>+}"
```

This rule is similar to regular expression, where "|" means "or", "<NN>+" means one or more <NN>. When given a sentence, it will check both patterns to see whether there is a match. For this sentence, it matches "<NN><IN><DT><NN>+". Then the phrase "size of a golf ball" will be extracted.

The full set of rules used to extract size information is shown in Table 4.4. The total number of rules in the grammar is five. Different tag patterns are used to capture different noun phrase structures.

Tag Pattern	Expression	Example
(<NN.*>+|<JJ>?<CD>?)<RP>?<TO> (<NN.*>|<VB><VBN>?|<JJ>+|<CD>) <IN>?<NN.*>*	... to ... size(d)	There was intense wind and some hail, **pea to marble size** with this storm.
(<NN.*>|<JJ>) (<IN><DT>?) (<VB.*>| <JJ>?<NN.*>+| (<CD>|<NN.*>)<CC> (<CD>|<NN.*>))	Size (of, between, from) ... (and) ...	Hail the **size of a golf ball**, very soft,flattened out on impact.
(<NN.*>|<JJ>)<CC>(<NN.*>|<JJ>) <VB.*>?<NN.*>+	... and ... size(d)	Tulia Jr. high also had .04 inch. However about 20 miles to our east and north, **golf and baseball sized** hail were reported at Happy and Dimmit.
(<NN.*>|<JJ>|<CD>+) (<VB.*><CD>?| <NN.*>)	... size(d)	5 minutes of hail at 12:30 PM. **Rice-raisin size**, mostly **pea size**. Cloudy and hard. will submit hail pad.
<JJ>+|<CD>	...-...	**Pea-sized**

Table 4.4: Regular Expression Grammar.

Besides the text description which contains the size information, there are still numeric descriptions which implicitly describe size information, such as integer, decimal, fraction. The

chunking approach we propose in this section just handles text description. Hence, we construct a rule-based method to extract size information for all reports, either in text description or numeric description.

4.2.2 Rule-Based Information Extraction

For this section, we propose a rule-based approach, which not only extracts the noun phrase information, but also numeric information like “1/4””, “1.25 inch”. The first step of our approach is to identify noun phrase information, and the second step is to identify numeric information.

4.2.2.1 Find Noun-Phrase (NP) information

Our method considers sentences, which have tokens in a keyword set, including “hail”, “hailstone”, “size”, “diameter”, “maximum”, “minimum”, “largest”, “smallest”, “larger”, and “smaller”. Then we search for the noun phrase of size shown in Table 4.2, like “pea”, “golf ball”, “tennis ball”.

Typos are very common for text typed by people. In addition, missing space between words is likely when they type fast. Thus, we use the edit distance, and select the terms which have less or equal to 1 edit distance with our terms in Table 4.2. One constraint of using edit distance is that it is limited to terms with length at least 5. The reason is that for short terms, it has a high probability to find terms within given edit distance in Table 4.2. For example, “sea” is within one edit distance from ‘pea’ , but “sea” is not a size term.

4.2.2.2 Find Numeric Information

Finding numeric information is more complex than noun phrases, because the number has different formats, like integer, decimal, fraction etc. In addition, there are many other kinds of numbers mixed with the size in the records. For instance:

- At 4:45 a violent storm with hail 1/4”, with occasional 3/8” to 1/2” pieces

In this example, there are different kinds of numbers, including the time, hail size, but we only consider the hail size as a valid number. Therefore, we define some rules to filter them out:

1. We consider numbers with unit “inch”, either followed by “inch”, “inches” or ”, like 1/4 inch, 2 inches, 1”.

2. If a number is not valid based on the first rule, but there is a proposition or conjunction, such as “to”, “and”, “or”, which connect it to another valid number, e.g. from 1/4 to 3/4 inch, we consider both of them as valid numbers.

Chapter 5: Experimental Evaluation

In this chapter, we present all the experimental results on two datasets, collected respectively from NCSCO and CoCoRaHS. We conduct text classification on NCSCO dataset, and information extraction on CoCoRaHS. Firstly, we describe evaluation metrics which are used in this Chapter.

5.1 Evaluation Metrics

There are various methods to measure effectiveness. Precision, recall, F_1, and accuracy are the most often used metrics. Precision measures the fraction of retrieved instances that are relevant, while recall measures the fraction of relevant instances that are retrieved. Those metrics are based on four numbers, including true positive (TP), false positive (FP), true negative (TN), and false negative (FN) (shown in the table 5.1).

	True condition positive	**True condition** negative
Predicted condition positive	True positive	False positive (Type I error)
Predicted condition negative	False negative (Type II error)	True negative

Table 5.1: Confusion Matrix

Precision is defined as:

$$precision = \frac{TP}{TP + FP} \tag{5.1}$$

Recall is defined as:

$$recall = \frac{TP}{TP + FN} \tag{5.2}$$

Accuracy is defined as:

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5.3}$$

F_1 score considers both the precision and the recall of the test to compute the score. It is commonly used as an indicator for comparing performances of different learning methods. F_1 is defined as:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{5.4}$$

5.2 Experiments of Weather Report Classification

In Chapter 4, we propose our approach to do weather report classification. Our approach is a combination of a rule-based method and a machine learning method. The rule-based method has a very high accuracy to assign a correct category to weather reports. Then a machine learning method (SVM) is used to deal with those weather reports which are hard for the rule-based method.

The dataset we use is the data collected by NCSCO. The number of records is about 1,000,000, spanning from year 2005 to 2014. We divide all records into two parts: records from 2005 to 2013 as training set, and the rest as testing set.

5.2.1 Analysis of Errors of the Rule-based Method

For the rule-based method, the accuracy is almost 99%. In order to further improve our results, we analyze some weather reports which the rule-based method cannot assign category based on "Event" attribute value of weather reports. There are mainly two cases below:

- **Empty event**: "Event" attribute has no value, it is blank.
- **Common event**: There are some events which fall into several categories. We call these events as 'Common Event". For example, "BLIZZARD" could be in the category "Snow", or "Storm Damage", or "High Winds". This way, there is a One-to-Many mapping, and we cannot decide which category should be assigned. Table 5.2 lists all three "Common Event" and their possible "Category".

For those two cases, we would like to categorize them based on machine learning methods.

Common Event	Category
AVALANCHE	Avalanche, Storm Damage, Snow
BLIZZARD	Snow, Storm Damage, High Winds
HURRICANE	High Winds, Storm Damage, Flooding

Table 5.2: Common Events

5.2.2 Weather Report Classification Experiment Setup

To classify those kinds of common events, we apply text classification methods with a preprocessing step. We concatenate "Event" and "Remarks" attribute values with ":" to form a new attribute "Description". "Description" attribute is the key attribute to predict category. Table 5.3 shows how "Description" is created based on "Event" and "Remarks" attribute values listed in Table 3.1, and the category for each record.

	Event	Remarks	Description	Category
Record 1	NON-TSTM WND GST	MESONET STATION BIG SABLE POINT /BSBM4/	NON-TSTM WND GST: MESONET STATION BIG SABLE POINT /BSBM4/	High Winds
Record 2	NON-TSTM WND GST	MESONET STATION MUSKEGON GLERL /MKGM4/	NON-TSTM WND GST: MESONET STATION MUSKEGON GLERL /MKGM4/	High Winds
Record 3	SNOW	24 HOUR SNOWFALL	SNOW: 24 HOUR SNOWFALL	Snow

Table 5.3: Concatenation of Event and Remarks

In order to use machine learning methods, we need to have a representation for each weather report suitable for the learning algorithm and the classification task. The first step in text classification is to transform documents, which typically are strings of characters, into a representation suitable for the learning algorithm and the classification task. Representing documents as a bag–of–words is a common technique in text classification. A free-text document is typically represented as a feature vector $\vec{d} = (x_1, \ldots, x_p)$, so that documents with similar content have similar vectors. Feature values x_i typically encode the presence of words, word n–grams, syntactically or semantically tagged phrases, Named Entities (e.g., people or organization names), etc. in the document. A standard method for computing the feature values x_i for a particular document d is called the bag–of–words approach. Each distinct word is a feature and the number of times the word occurs in the document is its value. This value is called the term frequency

$TF(w, d)$ of word w in document d. In addition, we do some post-processing.

- For the weather data used in this project, the text, just a few sentences, is much shorter than a usual document. It is very important to remove common stop words, such as "a", "the", "it", "is".

- When we represent a document as a vector of term frequencies, it is common to weight term values by different methods. There is a commonly used method in the domain of text classification, called TF-IDF term weighting scheme. The goal of using TF-IDF instead of the raw frequencies of occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus [11].

5.2.3 Classification Result

Our testing set has 125,106 records from year 2014. The vast majority of testing records are assigned correct categories through the rule-based method. We have 354 records which are assigned wrong labels. So we use machine learning method to train a prediction model to categorize those "hard" weather reports. The machine learning method we use is SVM. In order to evaluate our approach, we use the Naive Bayes method as our baseline. There are several Naive Bayes models that make different assumptions about how documents are composed from the basic units. We apply the two most commonly used models: Multinomial model and Bernoulli model. For this set of experiments, we use scikit-learn python library to train the Naive Bayes and the SVM models. For the setup, we use default experiment setting of each algorithm [5].

Table 5.4 shows the number of incorrect records and accuracy for each model. The precision, recall and F_1 score of each category for the three models are shown in Table 5.5, Table 5.6, and Table 5.7.

Model	**Incorrect records**	**Accuracy**
Multinomial Naive Bayes Model	42	88.14%
Bernoulli Naive Bayes Model	26	92.88%
SVM Model	5	98.59%

Table 5.4: Results of Three Machine Learning Models

	Precision	**Recall**	F_1**-score**	**Support**
Avalanche	1.00	0.18	0.31	11
Fire	0.00	0.00	0.00	1
Flooding	0.00	0.00	0.00	2
High Winds	0.80	0.13	0.23	30
Ice	0.00	0.00	0.00	3
Snow	0.88	1.00	0.94	304
Storm Damage	1.00	0.67	0.80	3
Avg / Total	0.86	0.88	0.84	354

Table 5.5: Multinomial NB Classification Results

	Precision	**Recall**	F_1**-score**	**Support**
Avalanche	1.00	0.18	0.31	11
Fire	0.00	0.00	0.00	1
Flooding	0.00	0.00	0.00	2
High Winds	1.00	0.70	0.82	30
Ice	0.00	0.00	0.00	3
Snow	0.92	1.00	0.96	304
Storm Damage	0.50	0.33	0.40	3
Avg / Total	0.91	0.93	0.91	354

Table 5.6: Bernoulli NB Classification Results

	Precision	**Recall**	**F1-score**	**Support**
Avalanche	0.92	1.00	0.96	11
Fire	1.00	1.00	1.00	1
Flooding	0.00	0.00	0.00	2
High Winds	0.97	1.00	0.98	30
Ice	1.00	1.00	1.00	3
Snow	1.00	0.99	1.00	304
Storm Damage	0.50	0.67	0.57	3
Waterspout	0.00	0.00	0.00	0
Avg / Total	0.98	0.99	0.99	354

Table 5.7: SVM Classification Results

Based on the above results, we can see that SVM performs better than the two baseline Naive Bayes model with respect to the final total F_1 and accuracy score. There are only five records are predicted incorrectly, which is shown in the Table 5.8.

No	Remarks	Predicted	Gold
1	avalanche: cars were stuck in floodwaters and mud on the columbia pass outside of sandy valley.	Avalanche	Flooding
2	hurricane: 5 ft storm surge msl in back creek	High Winds	Flooding
3	blizzard: at least six trees down near the vicinity of riverside road and piney forrest road in danville city.	Storm Damage	Snow
4	4-6 inches of flowing water on downtown streets.	Waterspout	Storm Damage
5	blizzard: trees down and road blockage reported east of community.	Storm Damage	Snow

Table 5.8: Incorrectly Classified Events

5.3 Experiments on the Magnitude Information Extraction

For the information extraction task, we propose two methods. Firstly, we come up with an NLP chunking method. The limitation of this method is that it can only extract noun phrase information. Secondly, we construct a rule-based method, which handles both noun phrase and numeric information extraction.

5.3.1 Experiment Setup

For this experiment, there are 47,409 CoCoRaHS weather records from year 1998 to 2015. Those weather records are all about the "hail" event. Among those records, there are 19,368 records with the word "size" mentioned, and 28,041 records that do not have "size". We use chunking method on the 19,368 records, and rule-based method on all 47,409 records.

We divide the records into two parts, including training and testing sets. The training set is used to find rules. For the experiment with chunking method, we randomly choose 1,500 records as the testing set. The other records belong to the training set. For the experiment with rule-base method, we randomly choose 1,000 records as our testing set. The rest are in the training set. For both testing sets, we do not have the gold magnitude. In order to measure the performance, we manually label their size information for both testing sets.

For both experiments, we use NLTK, which is a suite of libraries and programs for NLP, to pre-process text description. There are two common steps, including sentence segmentation and tokenization. For the chunking method, we also use Part-Of-Speech tagging techniques to get

the POS tag, which is important to define tag pattern.

5.3.2 Experimental Result

We present both experiment results of chunking based method and rule-based method in this section. For chunking experiment, we apply the grammar defined in Table 4.4 to our testing set, which consists of 1000 records. We extract correct noun phrase from 755 records. The accuracy is 75.5%.

Meanwhile, we apply the rules we define in Chapter 4.2.2 to all testing set. Table 5.9 shows the testing result for rule-based method in different data sets. As we can see from the second column of the table, the size of the testing set is 2500, and 1000 of those have word "size" , while 1500 of those do not have the word "size". From the third column, we can see that our rule-based method achieves 93.04% accuracy on all testing records. For the testing records with "size" information, it achieves an accuracy of 92.5%, performing much better than the chunking method does. For the testing records without "size" information, it also did very well.

	Total	Support (Accuracy)
With "size"	1000	925 (92.5%)
Without "size"	1500	1401 (93.4%)
Total	2500	2326 (93.04%)

Table 5.9: Rule-Based Testing Result

5.3.3 Analysis of Errors of IE methods

By sampling some incorrect records for chunking method, we summarize the error cases into the following categories:

- **Chunk incorrectly**

  There are other factors, such as text occurring position, affect the rules we created. For example:

  - heavy showers with pea sized hail 4/27 .

Based on the grammar shown in Table 4.4, we get the result shown in Figure 5.1:

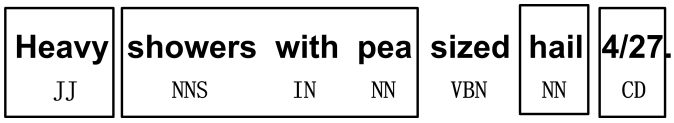

Figure 5.1: An Example of Incorrect Chunking

We can see that "showers with pea" is considered as a NP-chunk instead of "pea sized hail". The reason is that the rules cannot match overlapping noun phrases. For this case, a rule matches "showers with pea". Then other rules can just match the text after "showers with pea". So we can't find an NP that has word "size".

- **Can't cover all records.**
  There are records without the word "size" but still have size information, like numeric information. For example:

  - Hail at 9:20 pm, lasted 4 mins Stopped about 45 secs, restarted hailing again for another 3.5 mins (heavier hail). **Pea** *to* **marble***, plus several* **quarter-size***.*
  *- Hail between 6:43-6:49 p.m. 5/28; stone size:***1/4",1/2,3/8,5/8***, and* **3/4"***, mixed with rain*
  - Hail **1/4** *to* **1/2** *in diameter from 2:44 to 2:46.*

  Our rules do not cover those sentence structures, which occur very rarely in our dataset.

We also summarize the error cases into different categories for rule-based method:

- **Can't extract numbers without unit.**

  - Hail 14:14 to 14:23. Size between **1/8** *and* **1/4***, mostly the latter*

  According to our rules, the size information must follow unit "inch". The reason why we have this constraint is that there are many numbers in the records, like date, time. If we relax this constraint, there will be a lot of false positive records.

- **Tokens are not parsed completely**

  - 6/20 light hail began 1650-1705 **1/4-1/2inch***.'*

For this record, we should return [1/4, 1/2], however, “1/2inch” is not considered as a number because they’re concatenated together.

- **Can’t distinguish hail size from the precipitation depth or other events**

  - Smaller than **Pea** *size hail 1”- 2” deep.*
  - **nickel** *sized hail, 0.41 inches of rain.*

  For the above two examples, we not only find “pea” and “nickel” size, but also “1”, “2”, “0.41”. However, those numbers are not hail size.

- **Can’t find size not in our size dictionary**

  - precip was from rain and some beebe size hail that occurred at 16:15.

  If the reporter enters some size term not in the dictionary (Table 4.2), then we can’t extract it.

Chapter 6: Conclusions and Future Work

In this project, we developed a weather data system which consists of two components, weather text classification and information extraction.

For text classification, we developed a method based on a rule-based method and a machine learning method. The rule-based method can achieve 99% accuracy by doing weather event lookup in a predefined dictionary. For those records which are not assigned correct labels, we used machine learning methods to train a prediction model. We used SVM and Naive Bayes (Bernoulli and Multinomial) to train a classifier. From experimental results in Chapter 5, we find that SVM performs better than Naive Bayes. By combining rule-based and machine learning methods together, our method can achieve almost 100% accuracy over all event types.

For information extraction, our goal is to extract the magnitude information of hail event. We developed two methods, a chunking method, and a syntactic rule-based approach. For the chunking method, we mainly create a set of tag patterns based on POS tags. Those patterns are used to capture the structure of "size" noun phrases. For the rule-based method, we define different rules for extracting the noun phrase and numeric size information. From experimental results in Chapter 5, we find that the rule-based method performs much better than chunking does.

Our proposed approach is mainly rule-based. The limitation of this kind of method is that it requires a lot of domain knowledge, and is highly customized for different domains. Hence, for the future work, we consider constructing a more general IE system, which is not limited to one field but is applicable to many domains.

Bibliography

[1] Cocorahs. http://www.cocorahs.org/. Accessed: 2015-09-23.

[2] Information extraction. https://en.wikipedia.org/wiki/Information_extraction. Accessed: 2015-09-15.

[3] Information extraction techniques. https://opus4.kobv.de/opus4-fau/files/54/chapter03.pdf. Accessed: 2015-09-20.

[4] Named-entity recognition. https://en.wikipedia.org/wiki/Named-entity_recognition. Accessed: 2015-09-15.

[5] scikit-learn. http://scikit-learn.org/. Accessed: 2015-10-23.

[6] State climate office of north carolina. http://climate.ncsu.edu/office/about.html. Accessed: 2015-09-23.

[7] L. H. Lee K. Khan A. Khan, B. Baharudin. A review of machine learning algorithms for textdocuments classification. *Journal of Advances Information Technology*, 1, 2010.

[8] Stanley A. Changnon, David Changnon, E. Ray Fosse, Donald C. Hoganson, Richard J. Roth Sr, and James M. Totsch. Effects of recent weather extremes on the insurance industry: major implications for the atmospheric sciences. *Bulletin of the American Meteorological Society*, pages 425–431, 1997.

[9] Hamish Cunningham. Information extraction: A user guide (revised version). Department of Computer Science, University of Sheffield, 1999.

[10] Makoto Iwayama and Takenobu Tokunaga. Cluster-based text categorization: A comparison of category search strategies. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 273–280, 1995.

[11] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 143–151, 1997.

[12] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. 1998.

[13] David E. Johnson, Frank J. Oles, Tong Zhang 0001, and Thilo Gtz. A decision-tree-based symbolic rule induction system for text categorization. *IBM Systems Journal*, 41(3):428–437, 2002.

[14] V. TAMPAKAS M. IKONOMAKIS, S. KOTSIANTIS. Text classification using machine learning techniques. *WSEAS TRANSACTIONS on COMPUTERS*, 4(8):966–974, 2005.

[15] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48, 1998.

[16] Tom Mitchell. *Machine Learning*. McGraw Hill, 1996.

[17] Vincent Ng. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1396–1411, 2010.

[18] Jakub Piskorski and Roman Yangarber. Information extraction: Past, present and future. In *Multi-source, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing, pages 23–49. Springer Berlin Heidelberg, 2013.

[19] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[20] Yiming Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. pages 13–22. Springer London, 1994.

[21] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.

[22] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. pages 412–420. Morgan Kaufmann Publishers, 1997.

[23] Yuan F. Zheng Yi Liu. One-against-all multi-class svm classification using reliability measures. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks(IJCNN)*, pages 966–974, 2005.