A Cascaded Approach to Text Normalization for Email Data Cleaning

Aye Aye Theint, Myint Thu Zar Tun
University of Computer Studies, Maubin
ayeayetheint15387@gmail.com, myintthuzartun@gmail.com

Abstract

Email is one of the commonest modes of communication via text. By using email, people are sending and receiving many messages per day and communicating with partners and friends. Most of email data is very noisy. Thus, text normalization is the most popular and it is necessary to clean up email data. Text cleaning and normalization is a significant aspect in developing many text processing and information extraction applications in email data cleaning processes. Many text normalization applications need to take email as input. Text normalization has many methods to find the useful information. Among these methods, a Cascaded Approach is very suitable for cleaning email data. Our proposed system is to convert the canonical form from the "informally inputted" text by using text normalization. Moreover, this paper is to eliminate "noises" in the text and to detect paragraph and sentence boundaries in the text.

Keywords: text normalization, email data cleaning, information extraction, canonical form

1. Introduction

The World Wide Web (WWW) is an evolving system for publishing and accessing resources and services across the internet. Users use the web to retrieve and view documents of many types and interact with one another by email. Email has met tremendous popularity in the internet. Many text normalization applications need to take emails as inputs for email analysis, email filtering, and email summarization and information extraction from emails. In email, more and more informal text data becomes available to natural language processing such as raw text data.

Informally inputted text data is usually very noisy and is not properly segmented. Informal text may contain extra line breaks, extra spaces and extra punctuation marks and it may contain words badly cased. Moreover, the boundaries between paragraphs and the boundaries between sentences are not clear. In order to perform, high quality natural language processing, it is necessary to perform normalization on informally inputted data and then to remove extra line breaks, delete extra spaces, delete extra punctuation marks, delete unnecessary tokens and correct misused punctuation marks and restore badly cased words etc.

Email data cleaning is proposed in a "Cascaded" fashion. Email data cleaning is formalized as non-text block filtering and text normalization. Email data cleaning is defined as a proposed of eliminating irrelevant non-text data (which includes header, signature) and then transforming informal text data into canonical form at paragraph, sentence and word levels (text normalization).

This system aims to increase the knowledge of email data cleaning. Next, this system defines to understand text normalization on email. Then this system proposes to perform and formalize text normalization by using a Cascaded Approach. Finally, this system gives to understand about non-text block filtering and text normalization.

The rest of the paper is organized as follows. In Section 2 proposes related work. In Section 3 explains a Cascaded Approach in email data cleaning. In Section 4 describes the implementation of the system and in finally, Section 5 presents the conclusion and limitation.

2. Related Work

Text normalization is usually viewed as an engineering issue and is addressed in an ad-hoc manner. Much of the previous work focuses on processing texts in clean form, not texts in informal form. Clark investigated the problem of preprocessing. He proposes identifying token boundaries and sentence boundaries, resorting cases of words, and correcting misspelling words by using a source channel model in paper [3].

Lita et al. proposed employing a language modeling approach to address the case restoration problem. They define four classes for word casing: in all letters lower case, in first letter uppercase, in all letters uppercase and mixed case and formalized

the problem as that of assigning class labels to words in natural language texts in paper [4].

3. A Cascaded Approach in Email Data Cleaning

In this section, we present the detail of our system with some motivating examples. The system can only accept email which contains informally inputted text data. Users must extract information in email data. Then, users must transform this informally email data into the canonical form. Our proposed architecture is shown in Figure 1.

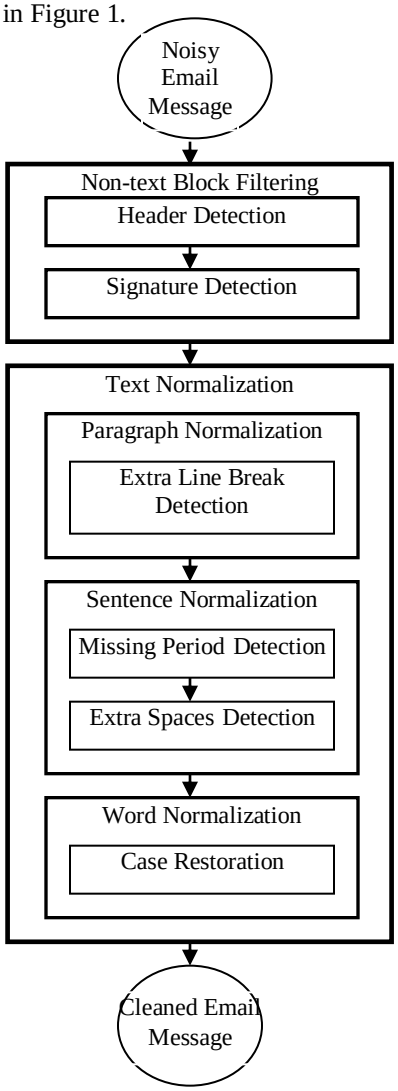

Figure 1. System Design of Email Data Cleaning

Our proposed system includes two parts; they are non-text block filtering and text normalization.

3.1. Non-text Block Filtering

The first step is non-text filtering step which includes header detection and signature detection. This step is essential for extraction information from email. Really these parts are not needed for email data so they must filter out as the non-text.

3.1.1. Header Detection

When an email message is sent to other people, it used to show that the mail is from whom, to whom, date, time and subject of the mail. The "From" field contains the sender's address of the message. The "To" field contains the address of one or more recipients who are the primary audience. The "Subject" field contains the title of the email message. Actually, "From", "To", "Date", "Time" and "Subject" are not real email data. They are not necessary for email data. Thus, they are assumed that non-text of the email data and then they are removed from email data.

3.1.2. Signature Detection

Email users also write their regard and respect at the end of the email. Really, they are not necessary for email data. They are also regarded as non-text and they are needed to filter out. The signatures may be such as "Regard", "Yours Sincerely", "Good Luck", and "Best Regards" and "Sincerely", "Thanks" etc.

3.2. Text Normalization

The second part is text normalization in email data. In this part, text normalization converts 'informally inputted' text into the canonical form, by eliminating 'noises' in the text and detecting paragraph and sentence boundaries in the text.

At a result of text normalization a text is segmented into paragraphs. Each paragraph is segmented into sentences with clear boundaries and each word is converted into the canonical form because there are dependencies between the processes. After normalization, most of the natural language processing tasks can be performed. One important activity at the text normalization phase involves paragraph normalization, sentence normalization and word normalization. Word normalization (example Case Restoration) needs

sentence beginning information. Paragraph normalization (paragraph ending information) helps sentence normalization.

3.2.1. Paragraph Normalization

In a document, a piece of text may contain many line breaks. User identify whether each line break is a paragraph ending or an extra line break and then users remove extra line breaks between paragraphs into a single paragraph. As a result, the text is segmented into paragraphs. Paragraph level has extra line breaks deletion and paragraph boundary detection. This step is mainly based on paragraph ending detection.

3.2.2. Sentence Normalization

One of the most important tasks of text normalization is sentence boundary detection or sentence splitting. Segmenting text into sentence is an important aspect in developing many texts processing application. . A sentence is a sequence of words ending with a terminal punctuation, such as a "." and "?". But, most sentences use a period at the end [2].

Sometime, a period can be associated with an abbreviation, such as "Mr.". In these cases, it is a part of an abbreviation, so user cannot delimit a sentence because the period has a different meaning here and it can be defined single a sentence break. However, an abbreviation itself can be the last token in a sentence in which case its period can be the last token in a sentence indicator (full stop).[1] .

If the users know that a capitalized word which follows a period is a common word, users can safely assign such period as a sentence terminal. Sentence normalization determines whether each punctuation mark (example period, exclamation mark and question mark) indicates at the end of the sentence. If the line starts "WH question", user adds question mark ("?") at the end of the sentence, otherwise, the line ends full stop (".").

It removes extra spaces in the sentence because single space needs between every word. So, many spaces are removed into a single space in every word. Moreover, it removes extra full stop at the end of the sentence into a single full stop. In sentence normalization, unnecessary token deletion refers to deletion of tokens like "-----", "=====", "*****", "#####" and "+++++" which are not needed in natural language processing. As a result, each paragraph is segmented into sentences. The sentence level is mainly based on the extra space deletion, extra punctuation mark deletion, missing punctuation mark insertion, misused punctuation mark correction, unnecessary token deletion and sentence boundary detection.

3.2.3. Word Normalization

Apart from being an important task of text normalization, the information about whether or not a capitalized word which follows a period is a common word allows users to accurately assign as sentence terminal. Normalization text from paragraph to sentences then words is also desirable, because there are dependencies between the processes. Word case restoration needs help from sentence beginning information and vice versa. Word normalization conducts case restoration on badly cased words.

Three possible types of casing for each word: in all characters lower case (AL), in first character upper case (FU) and in all characters upper case (AU). But, users do not consider a mixed case. In every sentence, small letter 'i' changes to capital letter 'I'. If the question mark is a sentence boundary, then the word after a question mark is capitalized. This case is called first character upper case (FU). Moreover, if the period is at the end of the sentence, a word after a period is capitalized. The day defines first character upper case (FU). And then, special words change in all characters upper case (AU). For example world wide web (**WWW**).

1. To :mgmg@gmail.com
2. Subject: How are you?
3. i am thinking about #### buying a Pocket
4. PC device for my wife this christmas….
5. the worry that i have is that she won't
6.
7.
8. be able to sync it to her Outlook Express
9. contacts.
10. how about you, too.what are you doing now.
11. i want to know your information?
12. Sincerely
13. koko

Figure 2. An Example of Informal Text

Figure 2 shows an example of informal text which includes many typical noises. Form lines 1 to 2 are a header, lines 12 to 13 are a signature. All of them are supported to be irrelevant to text normalization. Only lines 3 to 11 are the actual text context. However, the text is not in canonical form. It is mistakenly separated by extra line breaks. It also contains extra spaces between words "PC" and "device". Line 3 has unnecessary token such as "#####" to remove it. The first word in each

sentence (example “the”) should be capitalized. In line 4, there is an extra period after the word ‘christmas’. In every sentence, small letter ‘i’ changes to capitalize letter ‘I’. At the end of line 5, the line break should be removed in the sentence. The line 10 starts “WH question”, user adds question mark (“?”) at the end of the sentence, otherwise, the line ends full stop (“.”).

> I am thinking about buying a Pocket PC device for my wife this Christmas. The worry that I have is that she won’t be able to sync it to her Outlook Express contacts. How about you, too? What are you doing now? I want to know your information.

Figure 3. Formal Text by using Text Normalization

Figure 3 shows an ideal of output of text normalization on the input text in Figure 2. All the noises in the paragraph and sentence endings have been cleaned and identified.

4. Implementation of the System

Normalization from emails is an important subject in text normalization. Emails are usually noisy and simply applying text normalization tools to them, which are not designed for normalization from noisy data. Our proposed system based on email message as input. In Figure 4 illustrates which the user sends ‘informally inputted’ text to other user.

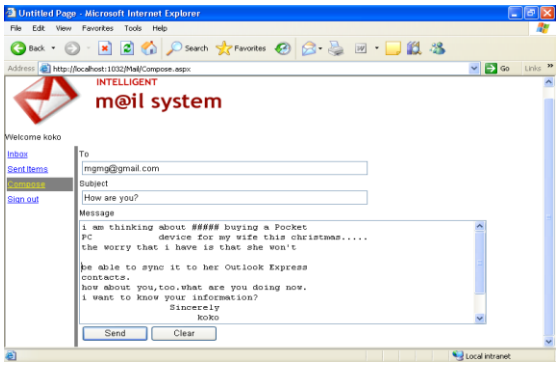

Figure 4. Informally Inputted Text

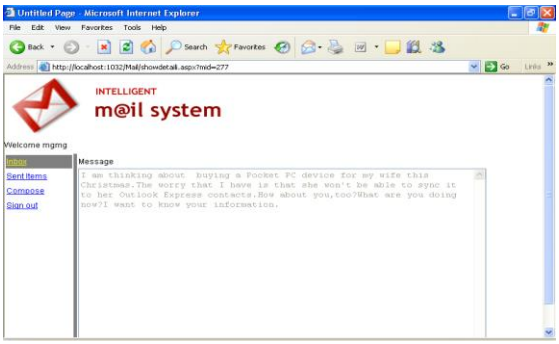

Figure 5. Ouput Text by using Cascaded Approach

After that, the propose system executes non-text block filtering which include header and signature detection and then text normalization that include paragraph, sentence and word normalization by using a Cascaded Approach. In Figure 5 shows that output of text normalization on the input text in Figure 4.

5. Conclusion

Email is the most common way in the internet and that is very important to transfer data. Email is needed to be flexible for every user. It is necessary to clean up unstructured text in the email data. Especially, email cleaning is defined as a process of eliminating irrelevant non-text data which includes header, signature and transforming relevant text data into the canonical form which contains paragraph, sentence and word normalization. We have proposed a Cascaded Approach to perform the task, especially to treat text normalization. Text normalization is an important issue for natural language processing. Natural language processing has many types of errors that are grammar errors, spelling errors and format errors. This paper can check only format errors. In making complete system for email data cleaning in which grammar errors can be checked in every sentence and spelling error can be checked in every word.

References

[1] A.Mikheev.2000.Document Centered Approach to Text Normalization, Proc. SIGIR 2000.

[2]D. D.Palmer and M.A Hear st.1997. Adaptive Multiling sentence Boundary Disambiguation, Computational Linguistics, and Vol.23.

[3]Clark, 2003.Pre-processing Very Noisy Text, Proc.of Workshop on Shallow Processing of Large Corpora.

[4].V.Lita. A. Ittycheriah, S Poukos, and N. Kambuhatla. 2003 tRuEasIng. Proc. of ACL2003

[5].Mikheev. A knowledge-free method for capitalized word disambiguation. In Proceedings of the 37th Conference of the Association for Computational Linguistics (ACL'99), pages 159-168. University of Maryland, 1999.