AKBL at the NTCIR-13 MedWeb Task

Reine Asakawa
Toyohashi University of Technology
asakawa@nlp.cs.tut.ac.jp

Tomoyoshi Akiba
Toyohashi University of Technology
akiba@cs.tut.ac.jp

ABSTRACT

The AKBL team participated in the Twitter subtask of the NTCIR-13 MedWeb Task. We tackled the task by using a machine learning technique, the Fisher’s exact test and real tweets which were collected under specific conditions. This paper outlines the methods we used to obtain the result evaluated by the task organizer.

Keywords

SVM, Fisher’s exact test, Data augmentation

Team Name: AKBL

Subtasks: Twitter (English, Japanese)

1. INTRODUCTION

The AKBL team participated in the Twitter subtask of the NTCIR-13 MedWeb Task [1].

Extracting tweets that mention actual influenza patients, from those related to influenza using SVM is researched by Aramaki et al. [2]. They described that even if tweets include mention of “influenza” or “flu”, they may not mention actual influenza patients. That suggests that not only disease/symptom expressions themselves but also other additional expressions appeared in their context play an important role to determin whether the one who post the tweet actually suffers from the disease/symptom or not. We thought that the amount of the tweets, which was distributed as a training data by the NTCIR-13, was insufficient to extract patient symptom information. For this task, we assumed that the reply tweet saying “get well soon” can be used as a clue to identify tweets indicating patient symptom information of some disease/symptom. From the collected tweets, we extract features for our classifier.

This paper is organized as follows. In section 2, we explain an overview of our proposed system, and we describe how to extract patient symptom information from the collected tweets. Section 3 explains the method for tweet classification. In section 4, we explain the classifiers used in the experiment and the result evaluated by the task organizer. Section 5 presents our conclusions.

2. PROPOSED SYSTEM

Figure 1 shows the configuration of our proposed system participated in the MedWeb task. The system employs eight binary classifiers using word features, each of which determines whether a tweet is positive for one specific disease out of eight or not. The eight classifiers share one set of features, referred to as “patient symptom words dictionary”. Each classifier for disease-X also has its own set of features, referred to as “disease-X words dictionary”. For the latter, we investigated two types of dictionaries. One is constructed from the labeled tweets provided from the task organizers, while the other is from general tweets whose labels are automatically obtained by applying the classifiers using the former dictionary.

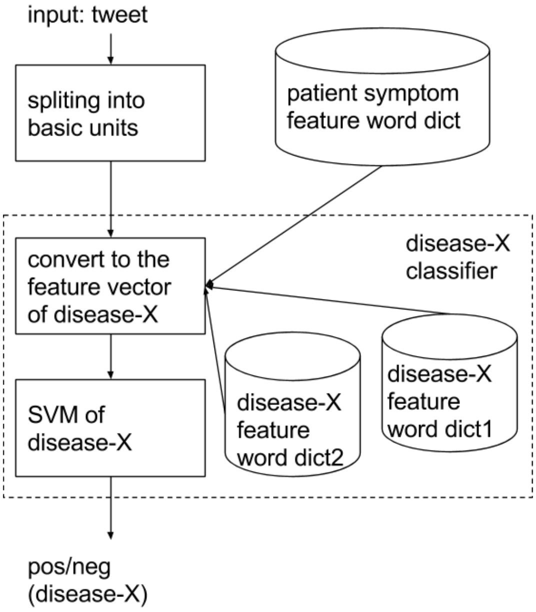
input: tweet
spliting into basic units
patient symptom feature word dict
disease-X classifier
convert to the feature vector of disease-X
SVM of disease-X
disease-X feature word dict2
disease-X feature word dict1
pos/neg (disease-X)

Figure 1: Our method

2.1 How to collect real tweets

We use two kind of tweets. The first is a set of general real tweets, which are collected during a certain period, in order to extract the patient symptom feature word. The second is a set of tweets that are replied by another tweet that contains the words “get well soon” or “お大事に”. We call that “Symptom tweets”. We think that they represent some symptom because “get well soon” is normally used for a person who complains of some illness.

We use tweepy, an API package for twitter of Python, to search tweets by keyword or tweet-ID and get the tweet-ID of the symptom tweet from a tweet that contains “get well soon”. According to subjective evaluation, about 70% of all of the symptom tweets represented some illness.

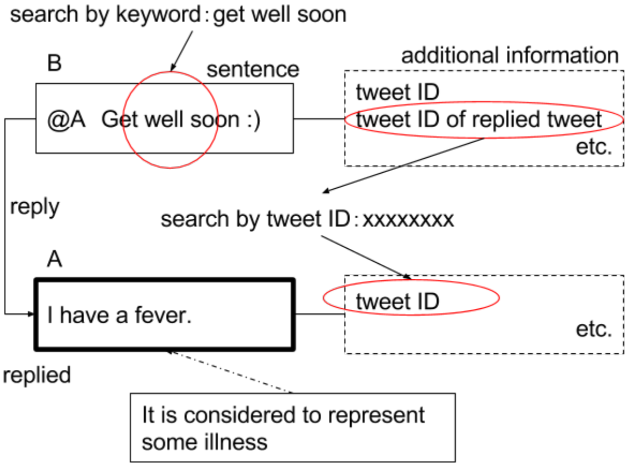
search by keyword : get well soon
B
sentence
@A Get well soon :)
additional information
tweet ID
tweet ID of replied tweet
etc.
reply
search by tweet ID : xxxxxxxx
A
I have a fever.
tweet ID
etc.
replied
It is considered to represent some illness

Figure 2: Symptom tweet

We found that there were two remarkable differences between Japanese and English tweets.

- The tweets said “get well soon” are not always in English but also in various languages other than English, while the tweets said “お大事に” are always in Japanese.

- The tweets saying “get well soon” are often retweeted by those other than their original authors. We had to remove such noisy retweets from our evaluation.

2.2 How to make feature words

We divide Japanese tweets into basic units by Morphological analyser, MeCab. We use mecab-ipadic-neologd as a dictionary of MeCab. This dictionary has a wide range of expressions. We convert English tweets into sequences of basic units by using an English POS tagger, TreeTagger. We also convert them into sequences of 3-grams using their surface form.

Japanese basic units are extracted by using MeCab (Example 1). Those contain surface words, parts-of-speech and lemma information.

Example 1

sentence: 私は日本人です.
basic units: [私, 名詞, 代名詞, 一般,*,*,*, 私, ワタシ, ワタシ], [は, 助詞, 係助詞,*,*,*,*, は, ハ, ワ], [日本人, 名詞, 一般,*,*,*,*, 日本人, ニッポンジン, ニッポンジン], [です, 助動詞,*,*,*, 特殊・デス, 基本形, です, デス, デス], [．,記号, 句点,*,*,*,*,．,．,．]

English basic units are made from results of TreeTagger (Example 2). The results of TreeTagger contain surface words, parts-of-speech and lemma information. We removed stop words from them. All contiguous sequences of three surface words (3-grams) are also extracted as basic units from a tweet (Example 3).

Example 2

sentence: I am Japanese. basic units: [I, PP, I], [am, VBP, be], [Japanese, JJ, Japanese], [., SENT, .]

Example 3

sentence: I am Japanese. basic units: [BOS-BOS-I], [BOS-I-am], [I-am-Japanese], [am-Japanese-.], [Japanese-.-EOS], [.-EOS-EOS]

For each basic unit, we create a 2x2 contingency table as shown in table 1. We try Fisher's exact test on the tables in order to select distinctive feature words from all basic units.

Table 1: 2x2 contingency table about “word”

	Replied	General	total row
“word”	n	m	$n+m$
other words	$N-n$	$M-m$	$N-n+M-m$
total column	N	M	$N+M$

Fisher's exact test is a statistical significance test used in the estimation of whether two factors in a 2x2 split table are independent or not. Although, the chi-square test is similar to this. However, when the sample size is less than 10, it is better to use Fisher's exact test than the Chi-square test. We implemented it with Python, stats.fisher_exact which is included in the Scipy package.

The calculated p value expresses a probability that the chance to find the basic unit in symptom tweets and that in general tweets are equally likely. We selected a word whose p value is less than 0.005 as the feature word.

The dictionary of feature words of patient symptoms was made by comparing 17,617 symptom tweets and 329,610 general tweets using Fisher's exact test and extracting the words that were used representatively in either the symptom tweets, or the general tweets.

The initial dictionary of feature words of 8-symptoms (disease-X feature word dict1) was made from the tweets distributed by NTCIR-13. It was made by comparing the tweets labeled positive for disease-X and all tweets by using Fisher's exact test, and extracting the words that were used representatively in the tweets that were labeled positive for disease-X.

Using those two dictionaries, we constructed eight SVM classifiers described later in Section 3. Then, they are used to classify 17,617 symptom tweets to obtain the pseudo labeled positive training data. From those tweets labeled (pseudo) positive for disease-X, we also constructed a second dictionary for eight symptoms (disease-X feature word dict2).

Example 4 is one of the feature words in the Japanese patient symptom dictionary. It consists of three elements: the result of morphological analysis parenthesised in square brackets, odds ratio, and p-value.

Table 2 shows the number of words in the each dictionary. Some of words are submitted in multiple dectionaries. The value about disease-X dictionary 1, 2 are average of eight symptoms.

Example 4

['寝', '動詞', '自立', '*', '*', '一段', '連用形', '寝る', 'ネ', 'ネ'], 6.07096899545, 6.86538886292e-288

Table 2: The number of feature words

	MeCab	TreeTagger	3-gram
patient symptoms dict	3620	1408	2263
disease-X dict1	33	21.9	40.1
disease-X dict2	12.4	11.1	-

3. CLASSIFICATION METHOD

Our classification method consists of the following steps.

1. Split a given input tweet into basic units.
2. Convert them into the feature vector for disease-X by using the patient symptom feature word dictionary and the disease-X feature word dictionaries.
3. Input the vector to a binary classifier trained for disease-X classification. Obtain a label of inputted tweet as an output.
4. Repeat steps 2 and 3 for each of the 8-symptoms, and save the results together finally.

We employed Support Vector Machines (SVMs) for the binary classifier [3]. RBF kernel was used for the SVMs. We defined the SVM parameter as $\gamma = 0.1$ and $C = 10$. We use the tweets distributed by NTCIR-13 as a training data. For the disease-X classifier, positive examples are ones labeled positive for disease-X, negative examples are the others. Moreover, we applied SMOTE to them because they were unbalanced data.

SMOTE (Synthetic Minority Over-sampling Technique) is one of the over-sampling approaches in which the minority class is over-sampled by creating “synthetic” examples rather than by over-sampling with replacement [4].

We implemented with Python, svm.SVC included scikit-learn package and over_sampling.SMOTE included imbalanced-learn package.

4. EXPERIMENTS

4.1 Six types of classifier

We submitted the result of the following three classifiers at the Japanese Twitter Tasks.

- (ja-1) use the patient symptom dictionary and both disease-X feature word dictionaries 1 and 2. The patient symptom dictionary and the disease-X dictionary 1 are used to select the words to be used as features of the classifier as they are. The disease-X dictionary 2 is used to see if the input tweet shares any words registered in the dictionary, then their number of unique types are used as the feature of the classifier.
- (ja-2) is almost the same as the above classifier. For this classifier, The disease-X dictionary 2 is used to see if the input tweet shares any words registered in the dictionary, then their total number are used as the feature of the classifier.
- (ja-3) use the patient symptom dictionary and the disease-X feature word dictionary 1.

We submitted the result of the following three classifiers at the English Twitter Tasks.

- (en-1) use the patient symptom dictionary and both disease-X feature word dictionaries 1 and 2. All dictionaries consist of uni-grams extracted by using TreeTagger. The patient symptom dictionary and the disease-X dictionary 1 are used to select the words to be used as features of the classifier as they are. The disease-X dictionary 2 is used to see if the input tweet shares any words registered in the dictionary, then their number of unique types are used as the feature of the classifier.
- (en-2) use the patient symptom dictionary and the disease-X feature word dictionary 1. All dictionaries consist of uni-grams extracted by using TreeTagger.
- (en-3) use the patient symptom dictionary and the disease-X feature word dictionary 1. All dictionaries consist of uni-grams and 3-grams.

4.2 Results

The Japanese and English experimental results are shown in Table 3 and Table 4, respectively.

The results show that the result of Japanese classifier is better than that of English. The poor performance on the English tweets seems to indicate that our feature selection method is not suited for English and that the quality of the symptom tweets of English is not as good as that of Japanese.

Table 3: Japanese result

	ja-1	ja-2	ja-3
Exact match	0.8	0.795	0.805
F1-micro	0.869	0.868	0.872
Precision-micro	0.889	0.891	0.896
Recall-micro	0.849	0.846	0.849
F1-macro	0.847	0.849	0.859
Precision-macro	0.873	0.875	0.883
Recall-macro	0.825	0.827	0.839
Hamming loss	0.030	0.030	0.029

Ja-3 showed the best performance among our three Japanese classifiers, even though it was the simplest classifier. Their vectors were created using patient symptoms dictionary and disease-X dictionary 1. Therefore, disease-X feature word dictionary 2 did not work well.

When comparing ja-1 and ja-2, ja-1 works better than ja-2 in terms of exact match and f1-micro. On the other hand, ja-2 is superior to ja-1 in terms of f1-macro. Their performances varied among the eight classifiers. Influenza classifier contributed most to improvement of the F1-macro

Table 4: English result

	en-1	en-2	en-3
Exact match	0.613	0.734	0.716
F1-micro	0.772	0.819	0.804
Precision-micro	0.656	0.832	0.853
Recall-micro	0.936	0.806	0.760
F1-macro	0.755	0.799	0.787
Precision-macro	0.649	0.808	0.834
Recall-macro	0.945	0.793	0.747
Hamming loss	0.065	0.042	0.043

of ja-2. However, it was found on only one tweet the difference of the result of Influenza classifier between ja-1 and ja-2. Such a small difference appeared as a big difference of F1-macro, because there were few positive examples in the test data. The Fever classifier of ja-1 was superior to that of ja-2 by four tweets. The difference of the result of the Fever classifier was bigger than that of the flu classifier, even it was opposite that the relationship in the difference of F values.

145 out of 640 tweets were classified incorrectly by one or more classifiers and 111 tweets were misclassified by all classifiers. We found that our classifier tended to misclassify the tweets whose reference labels were all negative. We also found that, while ja-1 and ja-2 shared many commons errors in their results, ja-3 did not with them.

En-2 showed the best performance among our three English classifiers, though it was the simplest classifier. Their vectors were created using patient symptoms dictionary and disease-X dictionary 1. Their basic units were made by only TreeTagger. Therefore, disease-X feature word dictionary 2 and the basic units made by 3-gram did not work well.

When comparing en-2 and en-3, en-3 is superior to en-2 in terms of precision, however, en-2 is superior to en-3 in terms of accuracy.

333 out of 640 tweets were classified incorrectly by one or more classifiers and 98 tweets were misclassified by all classifiers. We found that our classifier tended to misclassify the tweets whose reference labels were all negative.

5. CONCLUSIONS

We collected the real tweets and extracted the feature words using the Fisher ’ s exact test. We created two types of dictionaries from them. The initial dictionary has patient symptom feature words and the second dictionary has 8-symptoms feature words. We developed an SVM-based system using those dictionaries not only tweets distributed by NTCIR-13. Our method obtained exact matches of 0.8, 0.795, 0.805, 0.613, 0.734 and 0.716 at ja-1, ja-2, ja-3, en-1, en-2 and en-3, respectively. While our Japanese classifier relatively worked well, our English classifier did not work as well as it. The initial dictionary worked well. However, the second dictionary did not. For our feature work, we plan to use real tweets to train classifiers.

6. REFERENCES

[1] Eiji Aramaki, Shoko Wakamiya, and Mizuki Morita. Overview of the ntcir-13: Medweb task. *Proceeding of the NTCIR-13 Conference*, 2017.

[2] Eiji Aramaki, Sachiko Maskawa, and Mizuki Morita. Twitter catches the flu: Detecting influenza epidemics using twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP ’11, pages 1568–1576, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

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