A Highly Effective Hybrid Model for Sentence Categorization

Zhenhong Chen[1], Kai Yang[1], Yi Cai[1(✉)], Dongping Huang[1], and Ho-fung Leung[2]

[1] School of Software Engineering, South China University of Technology, Guangzhou, China
ycai@scut.edu.cn
[2] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China

Abstract. Sentence categorization is a task to classify sentences by their types, which is very useful for the analysis of many NLP applications. There exist grammar or syntactic rules to determine types of sentences. And keywords like negation word for negative sentences is an important feature. However, no all sentences have rules to classify. Besides, different types of sentences may contain the same keywords whose meaning may be changed by context. We address the first issue by proposing a hybrid model consisting of Decision Trees and Support Vector Machines. In addition, we design a new feature based on N-gram model. The results of the experiments conducted on the sentence categorization dataset in "Good Ideas of China" Competition 2015 show that (1) our model outperforms baseline methods and all online systems in this competition; (2) the effectiveness of our feature is higher than that of features frequently used in NLP.

Keywords: Sentence categorization · Hybrid model · N-grams · Feature

1 Introduction

Sentence categorization is a very important task in text analysis, because the types of sentences contain many useful semantic or syntactic information which can be used in many NLP applications, such as sentiment analysis [13] or question-answering (QA) systems [9], etc. The objective is to classify the type of a sentence as interrogative, negative or imperative and so on. Especially, the identification of negative and interrogative sentences attracts more attention. Because negative sentences express negative sentiment, while an interrogative sentence indicates interrogative attitude to specific parts of a sentence where QA systems need to analyze and give answers. There exist some grammar or syntactic rules to identify these sentences [12,14,18,25]. And keywords like "no", "not", "what", "when" also play an important role in determining sentence types [4,8,27]. However, there are still many sentences have no obvious rules to classify. In addition,

H. Gao et al. (Eds.): DASFAA 2016 Workshops, LNCS 9645, pp. 98–111, 2016.
DOI: 10.1007/978-3-319-32055-7_9

different types of sentences often contain similar keywords. For example, both of the interrogative sentence "Don't you play with us", and the negative sentence "I don't like playing games" have keyword "don't". Besides, many text corpora are from social media like microblog, or e-commerce sites, etc. These user-generated content (UGC) has a variety of informal expressions, which makes this task more challenged.

For this classification problem, there are various classifiers can be used, such as Decision Trees (DTs), Support Vector Machines (SVMs) [25] and so on. These machine learning algorithms are widely used. DTs are good at handling different decision rules and easy to interpret, while SVMs do better in classes that have no intuitive rules to classify. Because some sentences can be determined by rules while others don't have intuitive rules to judge, both of these two algorithms may perform very well on some sentences but not good on the others. In order to prove this assumption and improve classification accuracy, we propose a hybrid model which firstly classify sentences by DTs, and then use SVMs to handle those sentences hard to be judged.

As many machine learning applications, engineering an effective set of features is the main task of sentence cagorization. There are a diversity of features frequent used in natural language analysis, such as lexicons and their frequency, part of speech (POS), phrase position, etc. Among all of these, keywords are one of the most important features. However, different types of sentences may contain same or similar keywords. To enhance the effect of keywords, we design a new kind of feature based on the N-grams. This feature is generated by extracting the combination of keywords (like interrogative words or negative words) and POS of words in their N-grams, and calculating the occurrence probabilities of these combination in different types of sentence. This feature will be used in SVMs.

To validate the efficiency of our hybrid model and the feature, we use the dataset from the "Good Ideas of China" Competition 2015, sentence cagorization task. The dataset mainly contains text from social media. The hybrid model outperforms not only two baseline methods, but also all other systems of teams participated in this task. The quality of the designed feature is also evaluated by comparing it with other frequent-used features. It significantly improves the classification accuracy of SVMs in this task.

The main contributions of this paper are as follows:

- We propose a hybrid model aimed to efficiently determining the types of sentences.
- To the best of our knowledge, it is the first work to design the feature based on N-grams to apply for sentence cagorization.
- We conduct experiments on the dataset from "Good Ideas of China" Competition 2015, sentence cagorization task. As a result, our model outperforms the baseline algorithms and the other competition systems. Besides, the feature shows a high effectiveness compared with other features.

2 Related Works

As sentence categorization is a classification problem, there are several kinds of commonly used classification algorithms: Naive-Bayes (NB), k-Nearest Neighbors (KNN), DTs, and SVMs.

NB classifier is a probabilistic classifier based on Bayes' theorem [19]. It is a highly scalable algorithm which minimizes the probability of misclassification [7]. Different from NB, KNN is a non-parametric classification method [1]. In KNN, an object is assigned to the class most common among its k nearest neighbors.

DTs is one of the widely used approaches to multistage decision making, which uses a tree-like structure of decisions and their possible consequences [20]. For constructing DTs, we can use several commonly algorithm such as ID3 [23], C4.5 [17], ASSISTANT [15] and CART [3]. Among them, C4.5 is a quite popular algorithm which has been ranked 1 in the Top 10 Algorithms in Data Mining [24]. C4.5 is an extension of ID3 algorithm, and it use information entropy to build decision trees in the same way as ID3. Comparing with ID3, C4.5 made a number of improvements. It can handing both continuous and discrete attributes with differing costs [16] and dealing with missing attribute values. Therefore, we adopt C4.5 to generate DTs.

SVMs [6] is a supervised learning model used for classification and regression analysis. SVMs is one of the most robust and accurate methods among all well-known algorithms [24], and it is good at dealing with the problems such as nonlinear, high dimension. Given a set of training data, SVMs will search for a hyperplane by make margin between different classes as large as possible.

However, all the mentioned methods have some specific limitations. In order to balance the advantages and disadvantages of different algorithms, researchers propose different hybrid classification models. Kohavi et al. [11] use a Decision-Tree Hybrid Model to scaling up the accuracy of Naive-Bayes Classifiers. Khashei et al. [10] combine artificial neural networks and multiple linear regression models. Billsus et al. [2] propose a hybrid user model for news story classification. In this paper, we will present a hybrid classification model combing DTs and SVMs.

Negative and interrogative sentence identifications are two main tasks of text orientation identification. For negative sentence identification, Goryachev et al. [8] implement and evaluate four different methods of negation detection. Rowlett et al. [18] discuss the adverbials in negative sentence. Because we will apply our model on the Chinese dataset, we should also consider the characteristic of Chinese sentences. Zhu et al. [27] discuss the different between lexical negation, syntax negation, relative negation and absolute negation. Xu et al. [25] present a method based on semantic comprehension for text orientation identification. It utilizes SVMs to identify the text orientation, and find out those negative sentences. Yao et al. [26] propose a method to compute the sentiment orientation (polarity) of topics. Chen et al. [4] construct a Chinese negation and speculation corpus, then use the corpus to identify the negative sentences.

For interrogative identification, Ultan et al. [22] summarize some general characteristics of interrogative systems. Comorovski et al. [5] discuss the syntax-semantics of interrogative phrases. For Chinese interrogative sentences, Na-na et al. [14] explore the grammar mechanisms and characteristics of interrogatives to

determined whether they are used in interrogative sentences or not. Lan et al. [12] discuss the differences between interrogative uses and non-interrogative usages of WH-words. Shi et al. [21] analyze the exclamatory usages of interrogative words in Chinese sentences.

3 Methodology

3.1 Decision Trees

Decision tree (DTs) is a supervised learning model for multistage decision making problems. It is a tree-like structure that predicts the value of a target variable by learning decision rules inferred from the data features. In this structure, leaves represent class labels and branches represent possible decisions that lead to those labels. There are many commonly used learning algorithms, and C4.5 is one of most effective methods. So we use C4.5 to describe the building procedure of DTs. C4.5 uses information entropy ratio to build trees from a training set T with attributes A=$\{A_1,A_2,...,A_m\}$. It selects the attribute A_i with the highest information gain ratio g_R(T,A_i) to build the decision rule of each node from root to leaf:

$$g_R(T, A_i) = g(T, A_i)/H_{A_i}(T) \tag{1}$$

where $g(T, A_i)$ is the information gain and $H_{A_i}(T)$ is the entropy of T on attribute A_i:

$$g(T, A_i) = -\sum_{k=1}^{K} \frac{|C_k|}{|T|} \log \frac{|C_k|}{|T|} + \sum_{n=1}^{N} \frac{|T_n|}{|T|} \sum_{k=1}^{K} \frac{|T_{nk}|}{|T_n|} \log \frac{|T_{nk}|}{|T_n|}, \tag{2}$$

$$H_{A_i}(T) = -\sum_{n=1}^{N} \frac{|T_n|}{|T|} \log \frac{|T_n|}{|T|}, \tag{3}$$

where $|T|$ is the number of T's samples. Supposed T is split into N subsets T_n (n = 1,2,...,N), $|T_n|$ represents the number of the samples belonged to T_n. Supposed a decision tree has K classes C_k(k = 1,2,...,K), $|C_k|$ is the count of the samples that belong to C_k. And T_{nk} is the intersection between T_n and C_k.

DTs is an alternative technique for sentences categorization, because there are many effective attributes of sentences to help determine their types. For instance, interrogative sentences usually contain some fixed usages like "Do you", "Is it", "What are" and so on, while negative sentences have some structures like "don't" followd by verbs. So, if a sentence has one of these structures, it can be classified as an interrogative or negative sentence with a high possibility.

3.2 Support Vector Machines

Support Vector Machines(SVMs) are another kind of supervised learning model used for classification or regression problems, which have been shown to perform high efficiency at traditional text categorization. They aim to find the best decision hyperplane that separates data into different classes. In the two-category

case, the basic idea behind training process is to search a decision hyperplane, represented by $\overrightarrow{w}$, that not only separates the data of one class from those of the other class, but also maximizes the distance (i.e. margin) between two hyperplanes defined by support vectors; letting $y_i \in \{0,1\}$ be the correct class label of an input sample $\overrightarrow{x_i}$, the hyperplane can be written as follows:

$$\overrightarrow{w} = \sum_{i=1}^{n} \alpha_i y_i \overrightarrow{x_i} \tag{4}$$

where the α_i's value is obtained by solving the dual optimization problem, and n is the number of input samples. The $\overrightarrow{x_i}$ is a support vector of the hyperplane $\overrightarrow{w}$ if and only if the α_i is greater than zero. And the classification procedure is to decide which side of the hyperplane that input data fall in. Compared with decision trees, SVMs are more capable of handling non-linear classification by implicitly mapping input into high dimensional feature spaces with appropriate kernels like Gaussian kernel, etc.

3.3 N-gram

N-gram Model. N-gram model is one kind of probabilistic language models. It predicts the next word based on the previous (N−1) words in a sequence. This is a Markov model which assumes that the next item w_i depends only on the probability of the last (N−1) sequence $w_{i-(N-1)}^{i-1} = \{w_{i-1},...,w_{i-(N-1)}\}$, and the approximate prediction of a sequence w_1^n is made as following:

$$P(w_1^n) \approx \prod_{k=1}^{n} P(w_k | w_{k-(N-1)}^{k-1}) \tag{5}$$

An N-gram of size two is called bigram, and size three is trigram, and so on. Bigram and trigram models are mostly used.

N-gram Based Feature. As for our sentence cagorization, instead of using N-gram to train language models, we use it to design a new features of sentences. Supposed we have collected many keywords of interrogative or negative sentences like "what", "not" and save them into keyword lists, the new feature is generated as follows: firstly, we extract two kinds of combinations as follows (taking bigrams as an example):

$$(POS_{prev}, keyword) \; and \; (keyword, POS_{next}), \tag{6}$$

where POS_{prev} stands for the POS tag of keyword's previous word while POS_{next} represents the POS tag of the next word. And then, we calculate the occurrence probabilities of these combinations by their frequencies in different types of sentences. In short, the occurrence probabilities are the new kind of feature.

It is worth noting that we construct the N-gram based feature by POS, not by word itself. The reason is that POS may reflect deeper relationships between a keyword and its neighbors in a sentence. Because POS have a good ability to represent various classes of words. The combination of POS and keywords may contain more semantic or syntactic information. We will conduct experiments to explore the effectiveness of this feature. Furthermore, trigram features can also be constructed by the similar ways as the above process.

3.4 A Hybrid Model

In natural language, the types of many sentences can be determined by some grammar or syntactic rules. Because decision trees can efficiently construct decision rules and be easily interpreted, it is a very good model to use. However, there are still a diversity of sentences that can not be correctly judged only by these rules. There are two main cases. The first one is that a lot of sentences have no obvious rules, and their types mainly depend on semantic information. For instance, "Did I tell you?" is a interrogative sentence, while "Didn't I tell you that knocking on the door before entering the office?" is a rhetorical question. There are no direct rules to differentiate the types of these two sentences. Secondly, for corpora, especially the reviews from social media or e-commerce sites, there are many informal expressions lacking of complete syntax constituents. This makes sentence categorization more difficulty.

In order to solve these problems, we propose a hybrid model of DTs and SVMs, shown in Fig. 1 The basic idea is that DTs are firstly used to label sentences that can be determined by grammar or syntactic rules, and then the unlabeled ones are classified by SVMs, which learn decision hyperplanes between different classes by maximizing their margins. In addition, because features play an important role in machine learning algorithms, we design a new kind of features based on N-gram.

Hybrid Classification. As shown in Fig. 1, there are two main steps to predict the types of sentences from test set. First of all, DTs classify sentences by normal grammar or syntactic rules. For instance, a simple rule to determine negative sentences is that whether a sentence contains negative words like "no", "not". If it does not contain, then this sentence maybe classified into the branch of possible non-negative sentences; otherwise, it maybe classified into another branch of various possible negative sentences. And further decision would be made by next rules if existed.

When classification goes to a leave node of DTs, a sentence will either get a label of sentence types, or be unlabeled if it can not be judged by decision rules. For those unlabeled sentences, SVMs will determine their types. A diversity of features will be extracted to train SVMs. They use kernel techniques to search hyperplanes between different types of sentences, and make their margins as larger as possible to get better classification results.

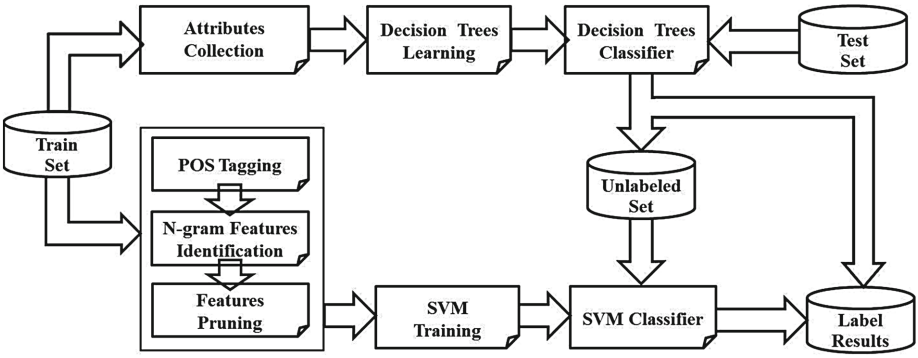

Fig. 1. A hybrid model

N-gram Based Features for SVMs. As most machine learning problems, the main task of sentences categorization is to engineer an effective set of sentences' features. So we design a kind of N-gram based feature for SVMs. As shown in Fig. 1, this feature is generated by three steps: keyword lists collection, POS tagging and N-gram features identification. To collect keyword lists, sentences of the train set are divided into different sets according to their class labels; word frequencies are counted in different sets and topN frequent words are selected into the corresponding keyword lists like negative keyword list or interrogative keyword list. To ensure the effectiveness, keywords of these lists will be manually selected. Besides, every word of sentences will be tagged with part of speech.

As described in Algorithm 1, keyword lists and sentences with POS are used to identify the N-gram based feature. After identifying the feature, its occurrence probabilities OP in every type t of sentences will also be counted (taking bigram based features as an example):

$$OP^t_{bigram_i} = \frac{Count(bigram_i \ occurs \ in \ S_t)}{Count(bigram_i \ occurs \ in \ S_{all})} \tag{7}$$

where $bigram_i$ is the i_{th} combination of the feature. S_t represents sentences of type t, while S_{all} represents all sentences in train set. The probabilities OP are the features used to train SVMs.

Last but no least, there are many other features can also be utilized to train SVMs, such as the length of a sentence, lexicons and their frequencies, punctuation and so on. We will conduct experiments to compare the efficiency between our feature and these frequent-used features, and engineer an effective set of features at last.

4 Experiments

We present a qualitative and quantitative analysis of the proposed hybrid model on sentences categorization task. And we also conduct experiments to evaluate the feature OP by comparing it with other frequent used features.

Algorithm 1. The generation algorithm of N-gram based feature OP

Input: Train set T; Keyword lists L_i; POS of words POS_w;

```
1  OP ← ∅
2  Set_{N−gram} ← ∅ //the set of OP's combinations
3  for each sentences s_k ∈ T:
4        do
5            for each keyword w_j ∈ L_i in s_k:
6                 do
7                      Set_{N−gram} ← (POS_{prev}, w_j)
8                      Set_{N−gram} ← (w_j, POS_{next})
9                      OP_{(POS_{prev}, w_j)} ← formula(7)
10                     OP_{(w_j, POS_{next})} ← formula(7)
11           end
12 end
```

Output: OP; Set_{N-gram}

4.1 Experiment Setup

Datasets Description. We conduct experiments on the sentences categorization dataset in the latest competition "Good Ideas of China", hosted by China Computer Federation (CCF). This dataset comes from the third task which aims to classify the Chinese sentences into three categories. Interrogative sentences and negative sentences are the first two categories, while sentences of other types are regarded as the third one. The train set and test set are directly provided to all participants.

Preprocessing. When looking into this dataset, we find that they are Chinese microblog text, and several preprocess operations are needed. First of all, text sequences need to be separated into independent sentences by Chinese or English punctuation. Secondly, there are some noisy content like nickname and URL needed to be removed. Thirdly, because Chinese words having no spaces between each other, we will use jieba Chinese module[1] to make segmentation and POS tagging. After having done these preprocess operations, we split the original train set (13,456 sentences) into a off-line train set, a development set and a off-line test set. The final distributions of the dataset are shown in Table 1.

Baseline Methods. We will compare our model with the following classification algorithms:

(1) *Decision Trees*: DTs are not only easy to interpret and explain, but also convenient to handle attributes interactions. The structure is directly based on decision rules, which are suitable for classification that can be relied on certain rules. They are non-parametric, and thus there is no a bunch of

[1] https://github.com/fxsjy/jieba.

Table 1. Statistics of "Good Ideas of China" competition 2015, Task3 dataset

	Interrogation	Negation	Others	Total
Train(off-line)	1149	658	7,693	9,500
Dev(off-line)	327	186	1,943	2,456
Test(off-line)	165	96	1,239	1,500
Test(on-line)	—	—	—	2,000

parameters to tune. We build it with C4.5 algorithm and it will give labels for all sentences even if some of them are very hard to classify for having no obvious grammar or syntactic rules.

(2) *Support Vector Machines*: SVMs are widely used baseline methods for natural language applications such as text categorization, sentiment analysis, and so on. They have nice theoretical guarantees and been shown high performance in various natural language tasks. We used Lin's (2011) libSVM[2] package with default RBF (Guassian) kernel for training and testing.

4.2 Model Analysis

For all models, we use the cross-validate approach to tune the parameters on the development set. According to the competition rules, precision P, recall R and F1 score F are defined as follows:

$$P = \frac{tp}{tp + fp}, R = \frac{tp}{tp + fn}, F = \frac{2PR}{P + R} \tag{8}$$

where tp is the number of interrogative and negative sentences which are predicted correctly, while fp is the number of sentences that are predicted as these two types but actually not; fn is the number of sentences which are these two types but not predicted correctly.

The proposed model and the baseline methods are firstly trained and tested on the off-line dataset. Table 2 shows the off-line P, R and F of these three models. From the comparison between two baseline methods, it is shown that DTs have a higher recall, while SVMs has a higher precision. The reason why decision trees have a high recall is that, instead of induct too specific rules that may cause over fitting, the decision rules tends to be suitable for various cases as many as possible, predicting more interrogative or negative sentences. However, there exist some cases hard to determined by rules. Take a sentence from the dataset as an example:

"你要去看电影不"
(Do you want to watch movie)

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvm/.

We may directly classify the English sentence as a interrogative sentence based on phrase "Do you". But in the Chinese sentence, there are no interrogative keywords or intuitive rules. Instead it contains a negation keyword "不" (no). So, when determined by general grammar or syntactic rules, the Chinese sentence would be misjudged as a negative sentence. The statistics of Table 3 shows that the misjudgement between interrogative and negative sentences is the main reason for lower precision of DTs.

Table 2. The P, R, F value of three models

Classifiers	Precision	Recall	F1 score
DTs	80.67	94.62	87.09
SVMs	**83.12**	89.88	86.37
Hybrid Model	82.96	**94.69**	**88.44**

Table 3. The P, R, F value of interrogative and negative sentences

Classifiers	Interrogative Sentences			Negative Sentences		
	Precision	Recall	F1 score	Precision	Recall	F1 score
DTs	80.71	95.91	87.66	80.59	92.34	86.07
SVMs	82.65	92.87	87.46	84.05	84.68	84.37
Hybrid Model	83.55	95.31	89.04	81.94	93.62	87.39

Besides, the reason why SVMs have a higher precision is that, instead of using hand-craft rules, this methods focus on directly maximize margin between various classes, and the hyperplanes are more accurate than the rules of DTs. On the other hand, because of the strict classification, SVMs predicted less interrogative or negative sentences and thus have a lower recall.

As shown in Table 2, our proposed hybrid model has the highest F1 score. It seems to combine the advantage of these two baseline methods. To make a deeper looking into this model, Table 4 shows the detail statistics about how these two methods contribute to the whole model's performance.

Table 4. Contributions of decision trees and SVMs

	DT Part	SVMs Part	Total
Precision	91.48	60.87	82.96
Recall	74.04	20.83	94.69
F1 score	81.84	31.04	88.44

DTs is the first part of the hybrid model to predict input, which is the main reason that they have a higher recall that is more than three times as that of SVMs. More importantly, as shown in Table 4, they have a very high precision which is bigger than that in Table 2. This is because, in the hybrid model, DTs only need to predict those sentences that can be judged by rules with high believe, while in the baseline methods, they must predict all sentences even if there don't exist correct rules to classify. Furthermore, for those sentences hard to correctly classified by rules, SVMs makes margin between different classes as large as possible. And because most rule-based sentences have been handled by DTs, SVMs can make a more fine-grained classification.

In short, the proposed hybrid model outperforms the baseline methods. It effectively handle most sentences which can be determined based on rules, and further make a more fine-grained classification for sentences which are hard to judged but have latent differences between various classes.

4.3 Features Analysis

For sentences difficult to judged by rules, we use SVMs to classify. As many machine learning applications, the main task is to engineer an effective set of features. In order to prove the efficiency of our N-gram features, we conduct experiments on a variety of frequent used features, as shown in Table 5.

Table 5. Features for SVMs in the hybrid model

	Sentence length	Lexicon	POS tag	Punctuation	Keywords count	Keywords position	N-gram features OP	P	R	F
1			√	√	√	√		60.98	19.37	29.40
2			√	√	√	√	√	60.68	20.70	30.86
3	√	√	√	√	√	√	√	**61.19**	20.33	30.52
4					√	√	√	60.87	**20.83**	**31.04**
5					√	√		49.38	19.95	28.42

Compared with the first experiment, the second experiment utillizes one more feature, our designed feature *OP*. As a result, the F1 score increases from 29.40 to 30.86, having an improvement of nearly 5 %. There are two possible reasons: (1) adding more features will improve performance, or (2) it is the effectiveness of *OP* that makes a difference. To check the first possible reason, two more features are added in the third experiment, but F1 score decreases from 30.86 to 30.52. Compared with the second experiment, its precision increases but its recall decreases, which indicates that too many features may cause over fitting. To solve this problem, the fourth experiment is conducted with only three parameter. Specially, F1 score increases up to 31.04. Most importantly, the comparison between it and the fifth experiment indicates that the proposed features is the main reason for the improvement of F1 score, which proves its high efficiency.

4.4 Compared with Online Systems

After we have evaluated the effectiveness of our hybrid model and the N-gram based features, we would like to apply our system for sentence categorization task of the competition, and compare it with the online systems of the competition. At the end of the competition, the results are as follows:

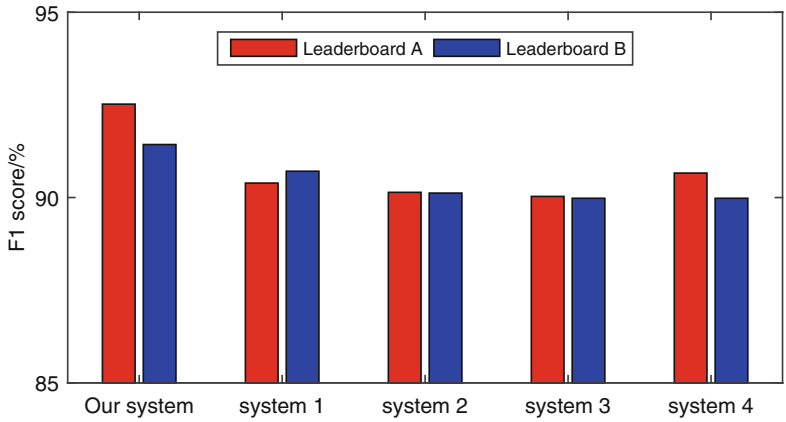

Fig. 2. Comparison with online systems

Figure 2 shows the top5 F1 scores of the sentence categorization task. The competition have two leaderboards: leaderboard A and leaderboard B, where A is shown during the whole process, while B is only shown at the last day. The results of these two leaderboards correspond to two test sets (each set has 1,000 sentences). The motivation is to test the robust of systems. Because leaderboards display name of teams instead name of systems, we use numbers to represent top5 systems. As is shown, our system has the best F1 score on both leaderboards, which indicates that our model is very robust and high effective.

5 Conclusion

We present a hybrid model by sequentially using DTs and SVMs to do sentence categorization. DTs determine types of sentences by rules, which is easily interpreted. SVMs classify sentences, which have not obvious rules to judge, by means of maximizing margin between various classes. On the other hand, one kind of feature is designed based on N-gram to make a better classification. Experimental results on the sentence categorization dataset of "Good Ideas of China" Competition 2015 verify the high effectiveness of our model and the feature. There are also several interesting directions for future research. For instance, N-grams based features may apply for other NLP applications like sentiment analysis. Moreover, sentence categorization can be used for further text mining.

Acknowledgments. This work is supported by National Natural Science Foundation of China (project no. 61300137), and NEMODE Network Pilot Study: A Computational Taxonomy of Business Models of the Digital Economy, P55805.

References

1. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. Am. Stat. **46**(3), 175–185 (1992)
2. Billsus, D., Pazzani, M.J.: A hybrid user model for news story classification. In: Kay, J. (ed.) UM99 User Modeling. CISM International Centre for Mechanical Sciences, vol. 407, pp. 99–108. Springer, Heidelberg (1999)
3. Burrows, W.R., Benjamin, M., Beauchamp, S., Lord, E.R., McCollor, D., Thomson, B.: Cart decision-tree statistical analysis and prediction of summer season maximum surface ozone for the vancouver, montreal, and atlantic regions of canada. J. Appl. Meteorol. **34**(8), 1848–1862 (1995)
4. Chen, Z.C.: Research on chinese negation and speculation identification. Master's thesis, Suzhou University (2014)
5. Comorovski, I.: Interrogative Phrases and the Syntax-Semantics Interface, vol. 59. Springer Science & Business Media, New York (2013)
6. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
7. Devroye, L., Györfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition, vol. 31. Springer Science & Business Media, New York (2013)
8. Goryachev, S., Sordo, M., Zeng, Q.T., Ngo, L.: Implementation and Evaluation of Four Different Methods of Negation Detection. DSG, Boston (2006)
9. Gupta, P., Gupta, V.: A survey of text question answering techniques. Int. J. Comput. Appl. **53**(4), 1–8 (2012)
10. Khashei, M., Hamadani, A.Z., Bijari, M.: A novel hybrid classification model of artificial neural networks and multiple linear regression models. Expert Syst. Appl. **39**(3), 2606–2620 (2012)
11. Kohavi, R.: Scaling up the accuracy of naive-bayes classifiers: a decision-tree hybrid. In: KDD, pp. 202–207. Citeseer (1996)
12. Lan, N.: Basic semantic study on wh-words. In: The Northern Forum, vol. 4, p. 013 (2005)
13. Liu, B., Zhang, L.: A survey of opinion mining and sentiment analysis. In: Aggarwal, C.C., Zhai, C. (eds.) Mining Text Data, pp. 415–463. Springer, Heidelberg (2012)
14. Na-na, T., Han, Y.L.: Mechanisms and characteristics of grammaticalization in interrogative words. Foreign Lang. Res. **5**, 016 (2009)
15. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986)
16. Quinlan, J.R.: Improved use of continuous attributes in c4.5. J. Artif. Intell. Res. **4**, 77–90 (1996)
17. Quinlan, J.R.: C4.5: programs for machine learning. Mach. Learn. **16**(3), 235–240 (2014)
18. Rowlett, P.: On the syntactic derivation of negative sentence adverbials. J. Fr. Lang. Stud. **3**(01), 39–69 (1993)
19. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Saddle River (1995)
20. Safavian, S.R., Landgrebe, D.: A survey of decision tree classifier methodology. IEEE Trans. Syst. Man Cybern. **21**(3), 660–674 (1991)

21. Shi, L.Z.: Exclamatory usages of question devices in contemporary chinese. Chin. Linguist. **4**, 14–26 (2006)
22. Ultan, R.: Some general characteristics of interrogative systems. Univ. Hum. Lang. **4**, 211–248 (1978)
23. Utgoff, P.E.: Incremental induction of decision trees. Mach. Learn. **4**(2), 161–186 (1989)
24. Wu, X., Kumar, V., Quinlan, J.R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Philip, S.Y., et al.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008)
25. Xu, L., Lin, H., Yang, Z.: Text orientation identification based on semantic comprehension. J. Chin. Inf. Process. **1**, 015 (2007)
26. Yao, T.F., Lou, D.C.: Research on semantic orientation analysis for topics in chinese sentences. J. Chin. Inf. Process. **21**(5), 73–79 (2007)
27. Zhu, Y.J.: Research on chinese language negation. Master's thesis, Tianjin Normal University (2012)