

COMSM0204 Report: Text Categorization

[anonymous] 17th April 2007

1 Introduction

Text categorization dates back to the 1960's but not until the beginning of 1990's did it become a major subfield of information systems discipline, thanks to the availability of powerful hardware. Since then, this subfield has gone through several stages of evolvments and even different branches of it such as spam filter have become hot research topics in the literature. Nowadays, text categorization is everywhere. Almost every industry which involves a certain degree of digital documentation process can be benefited from it.

This article is intended to give a brief introduction of the problem of text categorization and machine learning techniques that is related to it. The rest of the article is organized as follows. Section 2 gives a description of the problem, stating definitions, challenges and different types of text categorization tasks. Section 3 presents how digital documents are represented in order to go through classification process and various preprocessing techniques to enhance the performance of the system. Section 4 is about model building. The applications of several algorithms to text categorization are presented. Section 5 describes the evaluation methods of the models. Section 6 and section 7 presents topics from the literature.

OK.

2 The Problem

Text categorization (TC) is to assign a pre-defined category to natural language texts based on its content. TC used to be done by human experts to develop handcraft classification rules. This is very time consuming and as the amount of documents increases, it becomes infeasible. Automatic classification is proposed consequently. This is mainly done by using machine learning approaches which are to solve supervised learning tasks. To solve the problem of TC, there are some unique challenges. First, no matter what representation method is used, due to the vast amount of natural language words, a TC problem must have a great amount of features. Second, due to the amount of features and flexibility of representation of documents, preprocessing is very important to TC. The quality of preprocessing can have a big impact on the performance of building classifiers and the final generalization ability of the model. Third, because TC has a very wide application range, incorporating domain specified knowledge into different stages of solving the problem can play an important role. Different techniques that deal with these problems will be discussed later.

Different tasks may enforce different constraints to TC. These are some common types of TC tasks. *Single-label vs. Multi-label TC*: in single label cases, exactly one category will be assigned to one document. In multi-label cases, one document may have more than one category. Generally speaking, single-label techniques can be extended to multi-label cases.

Binary class vs. multi-class: Binary class case belongs to single label case. Multi-class problem can be dealt with in two ways. One is to use multi-class classifiers such as decision trees or do binary classification on a document on every category using a binary-class classifier.

Single layer classification vs. hierarchical structure categorization: In some applications such as web-page categorization, documents need to be assigned a sub-category under an upper level category. This can be accomplished by building a general model for main categories first and then different models for sub-categories.

OK

3 text representation and Preprocessing

Texts ^{Practically} cannot be interpreted by a classifier or a classifier building algorithm directly. The transforming process is usually called document indexing. As mentioned above, there are many ways to index a single document. However, different choices of indexing methods (representation) can have great impact on the building time and the generalizing ability of the model. Usually a document is represented by a vector of terms. In this case if 'term' means 'word', the representation is to regard each different word as a separate feature and whether a document process a word decided the value of the corresponding feature of that document. This value may be binary to indicate the appearance of the word in the document or an integer telling how many times the word appeared. This is often called 'bag of words' representation approach.

Of course 'term' can mean something else more sophisticated than single words, for example 'phrase'. However, in a number of experiments [1], it is revealed that these methods do not yield significantly better results. Lewis [2] argued that the reason for this is while 'phrase' indexing has superior semantic quality, it loses statistical information about the texts which word indexing contains. Apart from these two methods, there exist even more sophisticated representations that decompose sentences into hierarchical structure so to obtain maximum information including word, phrase, clause and even type of sentences. [There is also simplified 'bag of words' which disregard the position information of each word so only statistical information remains.] These representations serve for different purposes or are to fit particular assumptions or algorithms. A careful choice of representation is crucial for TC problems.

Besides representation, preprocessing is also very important. Preprocessing techniques can reduce the high dimensionality of the documents while keeping the necessary information to do classifications. Common preprocessing approaches including the follows.

Word stemming: the task of stemming is to reduce words to their stem, base or root form. For example the words 'thinker', 'thinking', 'thought', 'thinks' can all be stemmed to 'think'. Stemming sometimes need not to match the word to its exact base. A related base that has the same meaning will also be sufficient. Stemming can be done by using rules to match words to their bases.

Lemmatization: lemmatization can actually be categorized as one of a word stemming approaches. The difference between word stemming and lemmatization is the latter determines the part of the sentence in which the word exist before applying rules. Different parts of a sentence have different rules and the lemma might not be valid, but the semantics of the word formation is taken into account.

Stop words removal: this technique is to remove words that occur frequently but contains little information about what category the document should be assigned to, for example, 'the', 'this', 'of' and 'a'. Different stop word list is proposed and one can remove words that are above a certain frequency.

Feature selection: in order to further reduce dimensionality, feature selection is often applied to text vectors. Feature selection is to select features that can give greatest indication of which category the document should belong to and discard the other features. Some classifier algorithms cannot deal with great number of features, reducing feature number while keeping accuracy is also good for the training time of algorithms that can deal with mass feature classifications.

4 Building the model

Once the documents are ready, models can be built from them. Till now, a large number of classifiers have been applied to TC. Namely, they are decision trees, rule learners, nearest neighbor algorithms, neural networks, SVM, Bayesian classifiers, Rocchio method, RIPPER and sleeping-experts. As for most classifiers, the indexed and preprocessed documents can be applied directly.

Among all the classifiers which have been experimented with TC, SVM is most recently claimed to be of the highest accuracy [7]. There are some properties that why SVM is suitable for TC problems. *Higher dimensionality input*: dimensions of document representation can be up to 10000. SVM is an algorithm that does not depend on number of features so has the potential to handle large feature spaces. *Few irrelevant features*: in most high dimension feature space cases, one can omit most features as they are irrelevant to the problem. But in TC this is not the case. Usually a large number of features have considerable indication of where the document should end up. *Document vectors are sparse*: SVM has a bias that is suitable for dense concepts and sparse instances. *Most documents are linearly separable*: evidence indicated that most documents are linearly separable [7]. Support Vector Machines are to find these linear separations. Due to these reasons, SVM may be suitable for a large number of TC problems. Experimental results that showed SVM is more accurate than conventional classifiers are presented in [7].

However, SVM are not always the best. For example, when heavy interpretation of the classification model is needed, symbolic classifier such as decision trees are favored as while they can achieve good accuracy, they can also provide good explanation. The RIPPER and sleeping expert classifier are also different from the rest. While conventional classifiers usually disregard the context of the words, they take that into account. How the presence or absence of a word contributes to the classification of that document depends on the context of the word. The concept of context can be ambiguous and involves different criterion. But despite of this, these two algorithms present extremely good performance in many TC problems [4]. They are also good candidates in certain tasks.

5 Evaluation of the models

Evaluation of a categorization model employs measures from information retrieval in which originally TC is tackled with. Important measures are precision, recall and break-even point. Precision is the probability that a document which is classified as category C truly belongs to C. Recall is the probability that a document belongs to C is classified as C. Both measures are defined for binary classification problems. Between high recall and high precision there exists a trade-off. All classifiers can incorporate a thresholding confidence value to balance this trade-off. In order to evaluate multi-class problems, macro-averaging is proposed which averages the accuracy of different category queries in the document set. The precision/recall break-even point is defined as the threshold value for which precision/recall is equal. The evaluation often involves a confusion matrix which is similar to that of ROC analysis.

6 Feature Selection

Due to the importance of dimensionality reduction, lots of preprocessing methods has been proposed, among which feature selection plays the most important role. There exists many criterion of feature selection. A typical feature selection method computes a value for each feature.

Some summary
of what these
classifiers are
would have
been good
here.

Most classifiers
work on
linearly separable
data sets

More explanation

What does
this mean?

Explain how?

Reduction can be made by selecting certain number of features which have the highest values. 4 common feature selection methods are presented below.

Document Frequency(DF): document frequency is the number of documents in which a term occurs. Each unique term's DF is computed and terms whose frequency is less than some predefined threshold are removed. The assumption is that rare terms are less predictive and informative. Improvements is also possible if rare terms are noise. DF is the simplest method but there are arguments that low-DF terms should be more informative. ✓

Information Gain(IG): IG is the most popular feature reduction method. Based on entropy computation, it measures the bits information obtained if a term is presence or absence in a document. If the task has m categories, let c_i denote the i th category, the information gain of term t is defined as [3]:

$$G(t) = -\sum_{i=1}^m \Pr(c_i) \log_2 \Pr(c_i) + \Pr(t) \sum_{i=1}^m \Pr(c_i | t) \log_2 \Pr(c_i | t) + \Pr(\text{not } t) \sum_{i=1}^m \Pr(c_i | \text{not } t) \log_2 \Pr(c_i | \text{not } t)$$

but what does it mean? Why is high IG good?

This definition is specially designed for TC as it only takes presence and absence as value of t. Removal of terms is according to a threshold which will be compared with IG values of each term.

Mutual Information(MI): Mutual information is common in statistical language modeling of word associations. MI between term t and category c is defined as:

$$I(t, c) = \log_2 (\Pr(t \wedge c) / \Pr(t) * \Pr(c))$$

Let A be the number of times t and c occur together, B be the number of times t occurs without c, C be the number of times c occurs without t, and N be the number of total documents. $I(t, c)$ can be approximated by:

$$I(t, c) = \log_2 (A * N / ((A + C) * (A + B)))$$

All categories' scores of a term can be combined as:

$$I_{\text{avg}}(t) = \sum_{i=1}^m \Pr(c_i) * I(t, c_i)$$

MI favors rare terms as contradiction to DF.

χ^2 statistic(CHI): CHI compares the lack of independence between t and c. Let A,B,C have the same meaning as in MI and D be the number of times t and c neither occurs. CHI square between term t and category c is defined as:

$$\chi^2(t, c) = N * (AD - CB)^2 / ((A + C) * (B + D) * (C + D) * (A + B))$$

Similarly, the combined score of a term is:

$$\chi^2_{\text{avg}}(t) = \sum_{i=1}^m \Pr(c_i) * \chi^2(t, c_i)$$

The difference between MI and CHI is MI is normalized. But then CHI will not be reliable for low frequency terms. Why not? Upon which assumptions is the χ^2 test predicated?

In [3] evaluations of the above 4 along with Term Strength selection methods are presented. Reductions at all levels up to 98% removal are experimented with kNN and LLSF classifiers. IG and CHI is found to be most effective in aggressive reduction. DF thresholding is comparable in performance with the above two in reductions up to 90%. MI is inferior maybe due to its bias favoring rare terms. It is also found that DF, IG and CHI scores of a same term is highly correlated. Among them, DF is computationally cheapest. Comment on these claims ...

7 What is the best classifier for TC?

One may ask, in the many classifiers that have been applied to TC, which is the best? A lot of works has been done to compare classifiers. In [4], authors argued that RIPPER and the sleeping

expert out-perform all other classifiers. In [5] mentioned that kNN, LLSF and NNets (a neural network) are the best. In [6] and [7], SVM was claimed to be the most accurate. However, when looking at all these data, keep in mind that the data source they use to evaluate may not agree. Even a common collection is used. There are still many ways to introduce inconsistency. For example, Reuter's news stories are the most common text collection. It has at least 5 different versions depending on which subset of categories to use and how the training/test sets are divided. It has been known that the most serious problem in TC evaluation is the lack of standard data collection. In [1], a summation of all published performance of TC classifiers and systems is made. However we still cannot come to a final conclusion of which TC classifier is the best. At last keep in mind that the conservation of generalization rule tells us there is no single best classifier. ← Over ALL Different tasks need different consideration.

Over ALL
problems ; over all TC
Problems there maybe a
best....

8 Conclusions

Automated text categorization has become a very important aspect of information retrieval discipline. It is now heavily depending on machine learning approaches which automatically and inductively build classifier models. At the same time, TC has become a very challenging application to machine learning researchers. The properties of the tasks fit into a very typical machine learning problem. Thanks to the digitalization of documentations. It provides vast amount of resources for both studying and practicing the algorithms. ?

In a typical text categorization task, there are 4 technical stages. First is to choose a representation of texts. Second is to do preprocessing on the data. When the data is ready, existing algorithms can be applied to build models. The last stage is to evaluate the models. Due to the high dimensionality of TC problems, the first and second stages are of particular importance. Various methods are proposed to reduce the number of features while keeping the accuracy high. Among them feature selection plays the most important role.

Finally, there is still no confirmed best classifier for TC tasks. One has to evaluate the problem in a specific domain and test classifiers that meet the needs. ✓

Reference

- [1] F Sebastiani 'Machine learning in automated text categorization' ACM Computing Surveys (CSUR), 2002
- [2] LEWIS, D. D. 1992. 'An evaluation of phrasal and clustered representations on a text categorization task'. In Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval
- [3] Yiming Yang, Jan O. Pedersen, 'A Comparative Study on Feature Selection in Text Categorization', Proceedings of ICML-97, 1997
- [4] WW COHEN, Y SINGER, 'Context-Sensitive Learning Methods for Text Categorization', Conference on Research and Development in Information System
- [5] Y Yang, 'An Evaluation of Statistical Approaches to Text Categorization', Information Retrieval, 1999
- [6] S Dumais, J Platt, D Heckerman, M Sahami, 'Inductive learning algorithms and representations for text categorization'
- [7] T JOACHIMS, 'Text Categorization with Support Vector Machines: Learning with Many Relevant Features', Lecture notes in computer science

Comments

Your report does provide a brief introduction to the TC Problem, but would benefit from clearer explanations of the specialist terms used.

There's also not much critical assessment of the papers you've read, although you've correctly observed the fact that everyone thinks their classifier is the best!