Using Links to Aid Web Classification

Wei Xie Musa Mammadov John Yearwood
Center for Informatics and Applied Optimization
University of Ballarat
Victoria, 3353, Australia
Email: wxie@ncmail.com.au; m.mammadov, j.yearwood@ballarat.edu.au

Abstract

In this paper, we will present a new approach of using link information to improve the accuracy and efficiency of Web Classification. However, different from others, we only use the mappings between linked documents and their own class or classes. In this case, we only need to add a few features called linked-class features into the datasets. We apply SVM and BoosTexter for classification.

We show that the classification accuracy can be improved based on mixtures of ordinary word features and out-linked-class features. We analyze and discuss the reason of this improvement.

1. Introduction

With the rapid development of World Wide Web (WWW), extremely huge amount of information is now online and accessible by users. Automated classification on Web pages became more and more important and critical. In this paper, therefore, we will focus the text classification problems only on Web pages. This is also known as Web Classification.

Web pages are basically hypertext documents. These documents contain not only text (words) but also other contents, such as tags. In this paper, we will only focus on the useful information which is provided by the anchor tag, i.e. hyperlinks. Because of this special property of hypertext documents, Web pages can provide much richer information to text classifiers for classification.

Most researchers have used the text components of Web pages as the primary information for Web Classification. Other non-text components, such as meta-data or anchorwords, have been used to improve the accuracy of text classification results [1, 2, 8, 17]. The classification techniques [12], therefore, have been widely and successfully adopted and extended for Web Classification. There are many efforts have been done based on WebKB which is commonly used for Web Classification experiments [2, 6]. Study in [2, 13] showed that SVM based classification method produced better results than others.

2. Motivation

Web Classification has focused on text content of both target document and/or linked documents. The target document here means the document which is going to be classified. It is understandable that people are concentrating on text components of Web pages much more than the topological structure information which most Web pages provide naturally. Although some of them might consider the use of links or neighboring documents, they mainly concerned the text content of those linked documents. This kind of approach needs a huge amount of effort to extract all the text and special features of neighboring Web pages. The drawbacks of most of the current Web Classification methods are: 1) it needs a lot of extra storage space; 2) more computational power may be needed to cope with the much larger dimensions; 3) existing classification methods might need to be modified to satisfy the new requirements.

In this paper, we propose an approach which can improve the Web Classification accuracy without adding much effort, such as altering the existing classifiers or creating much larger databases for storing the external Web page information. Different from others, we only use the mappings between linked documents and their own class or classes. We call this kind of information *linked-class*. In this case, we only need to add a few features called *linked-class features* into datasets and apply different classification methods to them as usual.

Our approach for the use of link information for classifying Web pages was motivated by the following observations:

- Most Web pages usually contain URLs of Web pages (out-links) and are linked from external (in-links).

6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007)
0-7695-2841-4/07 $25.00 © 2007 IEEE

- People usually intend to link their Web pages to other Web pages that share similar or even the same topics, so these linked Web pages are likely in one or more common classes.
- All Web pages in a training dataset are assigned to one or more classes. We know the mappings between documents and classes. These pre-classified documents may be connected by a new document which needs to be classified. Therefore, we know about what kind of class or classes the target document is being liked or is linking to.

We can take advantage of linked-class information. By involving this information, Web Classification accuracy may be improved.

3. Dataset Preparation

3.1. Data Retrieval

In this work, we use the datasets generated from the intranet of the University of Ballarat. Using this allows us to have a dataset which has a relatively complete topological structure amongst the Web pages. Because we are trying to use the link information that exists in hypertext documents, only those hypertext documents within the university's intranet will be retrieved and stored.

3.2. Pre-processing

We randomly selected 2000 documents from the entire collection of 214,253 Web pages. Words in every document were extracted with the corresponding frequency. After stop-words removal [5, 15] and stemming [9], we have 4720 unique words left in our datasets. The links among these Web pages were also extracted. In this process, we only extract those inter-connected links among the 2000 Web pages.

These 2000 Web pages are manually assigned to 11 predefined classes accordingly. They are Services, Resources, Help, Policies, Plans, Announcement, Research, Teaching & Learning, Personnel, Finance and Advertisements.

3.2.1 Document Indexing

The Vector Space Model (VSM) [10] has been widely used in traditional text classification. In VSM, a document d_j is usually represented by a vector of feature weights

$$d_j = \left[w_{1,j}, w_{2,j}, \ldots, w_{|\mathcal{F}|,j}\right],$$

where $|\mathcal{F}|$ is the total number of features appear in the entire document collection, and $0 \leq w_{i,j} \leq 1$. $w_{i,j}$ is the weight of word i in document j. Hence, if $w_{i,j} > 0$, it means that word i exists in document j. For a set of features $\mathcal{F}$, there must be at least one feature i which has weight $w_{i,j} > 0$ in document j. Assembling feature vectors of all documents produces a documents matrix $\mathcal{W}$. The matrix is usually very sparse and the dimension of $\mathcal{W}$ could be very large.

There are four different ways to calculate weights for features, namely Binary Weights, Term Frequency (TF), Inverse Document Frequency (IDF) and TFIDF. We choose TFIDF [3] as the weighting system in this work. To limit a TFIDF value within the interval of 0 and 1, the value is usually normalized by cosine normalization [3],

3.2.2 Representing Link Information

Our approach is different from other Web Classification methods because we make use of linked-class information that exists in the topological structure of hypertext documents. In this case, therefore, we have to find a way to represent the link information in a document to make it ready for use.

As we mentioned earlier, every document is assigned to at least one class. Connected documents belong to some classes. These classes are called the *linked-class*. We count the number of neighboring documents, which belong to a particular linked-class, of a target document. The number is the frequency of the corresponding linked-class for the target document. The frequency of linked-class acts the same as TF.

- **In-linked classes**: Documents from outside point to the local document. We count the number of times a linked-class is used by in-linked documents.
- **Out-linked classes**: External documents are pointed by the local document. We count the number of out-linked documents that belong to each of these linked-classes.
- **Combined-linked classes**: Add in-linked classes with corresponding out-linked classes together. It means that, we still have 11 linked-class features in the datasets, but the values of each linked-class features are the sum of both in- and out-linked classes.

After that, we apply TFIDF weighting procedure onto the datasets to calculate weights for both ordinary features and the linked-class features.

3.3. Generation of Datasets with Link Information

In this paper, we are concerned with 3 different cases regarding the use of link information. They are in-linked-classes, out-linked-classes and combined-linked-classes.

Based on the original datasets that only contain the ordinary features, denoted as D, we put 11 extra linked-class features in to form new datasets. These extra 11 features represent the linked-classes information, the weights of each of them in different documents. We then name the new generated datasets DIn, DOut and DCom respectively.

For a given document d, we denote its class information by

$$c(d) = \left[(c_1(d), c_2(d), \ldots, c_{|C|}(d)\right]$$

where $c_k(d) = 1$ if document d belongs to class k; $c_k(d) = 0$, otherwise.

For DIn, consider a document d and assume that, $\{\widetilde{d_j}\}_{j=1}^{|N_{in}|}$ denotes the set of in-linked documents of d and $|N_{in}|$ is the number of in-linked documents of d. The in-linked-class information of document d_j is

$$c(d_j) = \left[c_1(\widetilde{d_j}), c_2(\widetilde{d_j}), \ldots, c_{|C|}(\widetilde{d_j})\right],$$

where $c_k(\widetilde{d_j}) = 1$ if document d_j belongs to class k, otherwise $c_k(\widetilde{d_j}) = 0$.

For this in-linked-class information, we consider a vector

$$in(d) = \left[in_1(d), in_2(d), \ldots, in_{|C|}(d)\right]$$

where

$$in_k(d) = \sum_{j=1}^{|N_{in}|} c_k(\widetilde{d_j}), \quad k = 1, \ldots, |C|. \tag{1}$$

The value of $in_k(d)$ is the frequency of linked-class feature k in document d. In other words, this value can be considered the same as the TF value of linked-class feature k. As we mentioned, we use TFIDF to weight each of the words in a document. After we calculate the TF values of linked-class features, IDF, another basic element of TFIDF is obtained by using the following methods.

For IDF values of the linked-class features in a document d, we consider a vector

$$in(idf) = \left[in(idf)_1, in(idf)_2, \ldots, in(idf)_{|C|}\right]$$

where

$$in(idf)_k = \sum_{j=1}^{|N|} a_k(d_j), \quad k = 1, \ldots, |C|, \tag{2}$$

and

$$a_k(d_j) = \begin{cases} 1 & \text{if } in_k(d_j) > 0; \\ 0 & \text{otherwise.} \end{cases}$$

After we calculated both TF and IDF values, we obtain TFIDF value of every feature in different documents by using

$$ilc_k(d_j) = \frac{in_k(d_j)}{in(idf)_k}, \tag{3}$$

and then normalize them.

Each document d, therefore, will be represented by a vector of features

$$d = \left[w_1(d), \ldots, w_{|\mathcal{F}|}(d), ilc_1(d), \ldots, ilc_{|C|}(d)\right],$$

where w_i represents the weight (TFIDF value) of ordinary word i in document d.

By putting all the documents together, we obtain a dataset DIn which contains both ordinary features and in-linked-class features for every document. To generate DOut and DCom we will apply the similar procedure. Moreover, for DCom, the in-linked and out-linked classes frequencies will be summed together, but the number of linked-class features should remain no change. Therefore, instead of having (1), we should have (4).

$$io_k(d) = \sum_{j=1}^{|N_{io}|} c_k(d_j) = \sum_{j=1}^{|N_{in}|} c_k(\widetilde{d_j}) + \sum_{j=1}^{|N_{out}|} c_k(\widetilde{\widetilde{d_j}}), \quad k = 1, \ldots, |C|. \tag{4}$$

3.4. Selecting Features Based on IG

Text datasets usually have large number of features. The dimensionality reduction, i.e. feature selection, is a necessary procedure in Text Classification. In this work, features were selected based on the Information Gain [16] values of every feature in different datasets. To achieve this, both ordinary and linked-class features, need to be ranked based on their IGs in different datasets. A list of the ranks of each linked-class features is given in Table 1.

Based on the rank of each feature, we generate sub-datasets of D, DIn, DOut and DCom that contain varying numbers of features from 100 to 1000 respectively by selecting the corresponding highest ranked features. This allows us to have more datasets for testing. Meanwhile, the number of linked-class features involved could vary. This gives us the chance to see how those linked-class features can affect the classification results.

Considering the 100-feature set, according to the ranks, DIn, DOut and DCom contain 93 ordinary features and 7 linked-class features. For example, in Din, linked-class feature 1, 2, 3, 4, 5, 9 and 10 will be selected into 100-feature set. This means that the 7 "least informative" ordinary features (i.e. the last 7 features in the 100-feature sub-datasets of D) will be pruned and be replaced with the 7 linked-class

	DIn	DOut	DCom
Linked-Class 1	23	13	19
Linked-Class 2	62	62	62
Linked-Class 3	67	73	69
Linked-Class 4	12	12	16
Linked-Class 5	53	50	53
Linked-Class 6	168	101	103
Linked-Class 7	259	430	296
Linked-Class 8	368	111	111
Linked-Class 9	9	9	9
Linked-Class 10	80	75	67
Linked-Class 11	264	170	175

Table 1. IG ranks for each of the linked-class features in DIn, DOut and DCom.

features. The ranking show that all linked-class features are informative if compared with the totally 4720 ordinary features.

4. Experiments Settings

4.1. Folding

In the numerical experiments, we will use 4-fold cross-validation. We note one fact which is related to linked-class features. That is, because all test sets are supposed to be unknown, we have to eliminate all links connected to the documents in test sets and keep only links connected to the training examples.

4.2. Classifiers

As mentioned, SVM is one of the most commonly used algorithms in Text Classification [14, 13]. In the experiments of ours, we use SVM^{light} [7] as it provides high performance in many applications. We also applied BoosTexter [11], another well-known and widely used algorithm, on our datasets.

4.3. Evaluation

In this paper, we will use the Average Precision measure considered in [11]. Note that, this measure allows us to achieve more complete evaluation in multi-label classification problems.

Because the text components have been considered as the primary information for Web pages, the simplest approach, also known as the Text Only approach, which is to use only text features in Web Classification [4], has been implemented widely. This approach can supply a baseline of classification performance for other methods that consider also other non-text components. For this reason, we applied the two classifiers to the dataset D and then measure the results to set our baseline.

5. Results and Analysis

The SVM based text classification results for all datasets in our work are listed in Table 2 and 3 (both training and test phases). In the following tables, F100 and F200 indicate the numbers of features in the corresponding datasets are 100 and 200 respectively, and so on.

	D	DIn	DOut	DCom
F100	82.24	82.23	**82.69**	82.47
F200	82.84	86.06	**86.49**	86.10
F300	87.73	87.54	**87.85**	87.58
F400	88.77	88.76	**89.20**	88.81
F500	90.01	89.88	**90.31**	89.78
F600	90.95	90.77	**91.17**	90.69
F700	91.58	91.39	**91.85**	91.37
F800	92.12	91.93	**92.22**	91.91
F900	92.14	92.10	**92.48**	92.12
F1000	92.17	92.20	**92.50**	92.21

Table 2. Results obtained by SVM for training sets.

	D	DIn	DOut	DCom
F100	76.53	76.58	**76.62**	76.55
F200	77.74	77.84	**78.35**	77.94
F300	79.10	78.16	**79.21**	78.35
F400	79.55	78.87	**79.91**	78.88
F500	79.67	78.83	**79.88**	79.15
F600	79.41	78.77	**79.89**	79.10
F700	79.69	79.18	**79.88**	79.21
F800	80.23	79.58	**80.25**	79.65
F900	80.15	79.75	**80.61**	80.10
F1000	80.42	79.70	**80.76**	81.80

Table 3. Results obtained by SVM for test sets.

We put bold numbers when an average accuracy is the best. By observing the tables, therefore, we simply found that, for both training and test phases, DOut provides the best results for most cases; DCom and DIn provide no better results than DOut and sometimes the results are not as good as the results from original datasets D. To make it clearer,

we also generated the overall average accuracy for each of the data types. It is shown in Table 4.

	D	DIn	DOut	DCom
Training	89.06	89.29	**89.68**	89.30
Test	79.25	78.72	**79.54**	79.07

Table 4. Overall Results obtained by SVM.

The results from BoosTexter are shown in Tables 5 and 6. BoosTexter at the beginning, however, did not work with the datasets as well as SVM did. From the BoosTexter results, we can notice that, out-linked-class features worked very well in the training phase for all experiments. In the test phase, when the size of the feature set equals 100 or 200 and some others 600, 800, the out-linked-class features produced better results. This simply means that out-linked-class features can improve the performance of BoosTexter.

	D	DIn	DOut	DCom
F100	88.08	87.97	**88.59**	88.04
F200	93.87	94.06	**94.15**	94.26
F300	96.61	96.99	**97.20**	97.04
F400	97.25	96.46	**97.60**	97.60
F500	97.50	97.83	**97.94**	97.83
F600	97.73	97.97	**98.02**	97.98
F700	97.83	98.04	**98.21**	98.05
F800	97.90	98.40	**98.28**	98.21
F900	97.94	98.26	**98.33**	98.20
F1000	98.22	98.23	**98.30**	98.33

Table 5. Results obtained by BoosTexter for training sets.

	D	DIn	DOut	DCom
F100	78.78	79.22	**79.64**	78.75
F200	80.19	79.71	**80.24**	79.69
F300	81.17	79.40	80.62	79.92
F400	81.86	80.45	81.66	79.75
F500	81.23	80.31	81.11	79.30
F600	80.84	80.00	**81.46**	80.17
F700	81.33	80.70	80.60	80.79
F800	81.09	80.28	**81.22**	79.58
F900	81.83	80.21	81.13	79.90
F1000	81.10	80.03	80.64	80.77

Table 6. Results obtained by BoosTexter for test sets

Looking at Table 6, for example, DOut that contains 200 features obtained better accuracy than original dataset D. This means that out-linked-class features are informative and improved BoosTexter's accuracy. According to Table 1, in 200-feature sets, 10 out of 11 out-linked-class features have been selected into the datasets, these out-linked-class features show no change until 500-feature sets. In the 300- and 400-feature sets' cases, BoosTexter did not work as well as it did with the 100- or 200-feature sets. This however would not be expected to happen, unless some of the newly added-in features are NOT informative. These features are ***ordinary features***, rather than out-linked-class features.

With the above analysis in mind, we decided to implement another set of experiments to test our analysis. We notice that the out-linked-class features in the 200- feature sets are the same as those in the 300- and 400-feature sets. The difference between those datasets is that we newly added some ordinary features into the datasets. Because no linked-class features were changed, this means that the newly added ordinary features had some possible side effect on the results. Therefore, we believe that the ten excluded (by 200-feature sets) ordinary features, namely sub-set A, may be less informative than the ten ordinary features, namely sub-set B, which were excluded by 300-feature sets. We then replaced the sub-set A with sub-set B. In other words, we replaced less informative features with potentially more informative features. We applied BoosTexter on the new datasets (300 features). We present the results in Table 7.

	D	DOut (A)	DOut (B)
Training	96.61	97.20	97.12
Test	81.17	80.62	81.69

Table 7. Results from BoosTexter for 300-feature DOut with sub-set A in it and with sub-set A replaced by B.

As you can easily discover, when BoosTexter worked with DOut which contains sub-set B instead of sub-set A, we obtained the best average precision out of other datasets. Our approach in the new implementation worked again.

Regarding to this issue, we note that IG measure is not a perfect method for choosing the "most" informative features. In other words, the rankings shown in Table 1 does not necessarily mean that, higher ranked features are more informative than lower ranked ones.

From observing the results that we obtained by using both SVM and BoosTexter algorithms, DOut obtains the best average accuracy in these 4 different types of datasets. This raised another question: Why does DOut gets the best results? To know the reason we must analyze the creation and use of out-links.

In-links are the links that point into a Web page from outside; reversely, out-links are the links point out to other Web pages in WWW networks. Therefore, the out-links are

more controllable by a Web designers, but the in-links are different. This means that the purpose of an out-link is more specific than an in-link. When a Web designer points the target document to some other documents, this means that the topic or content of the base document is related or close to the out-linked documents. Therefore, the possibility that they belong to the same classes is higher. However, in-links can come from all kinds of sources, it is unpredictable. Although the in-linked documents may share some common topics as the local document, how much they are similar to each other is unsure.

One may think that an in-link must correspond to an out-link, so the in-links and out-links should give the same effects. However, this is not true. For example, look at the internal home page of the University of Ballarat, it out-links to so many other Web pages in various topics. It seems that the out-links should produce worse result. Yes, that is true. However, this is only for some particular kinds of Web pages and the number of such documents is tiny compared with the number of all Web pages in a network. A few wrong classification would make no big difference to classifications accuracy. Now, we think about this problem from those out-linked pages of the internal home page. We will see that the internal page adds 1 to every linked-class (which it belongs to) feature for its out-linked Web pages, so one document affects many other documents among the entire data collection. It is obvious that the bad effect from in-links on each of the documents is more likely greater than out-links. On the other hand, out-links from those documents may not point back to the internal home page at all. Therefore, out-linked-class information should be more accurate.

Thus, when we added those out-linked-class features into the datasets, we added some more informative features than most of the ordinary features existing in the datasets; when we added other types of linked-class features in, the situation is uncertain. According to the above analysis, DOut should get better results than others.

6. Conclusion

In this paper, we showed a simpler way to make use of link information to aid the Web Classification procedure. Unlike most other Web Classification approaches, our method does not use the text component of linked documents. Our approach is to use the mappings between linked document classes to give positive effects on Wext Classification. It is more effective in term of average precision. With this approach, we showed that using out-linked-class features improves the classification accuracy. Meanwhile, out-linked-class features outperformed other types of linked-class features that we considered in this paper, in-linked and combined-linked.

References

[1] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. *SIGMOD Rec.*, 27(2):307–318, 1998.

[2] M. Craven and S. Slattery. Relational learning with statistical predicate invention: Better models for hypertext. *Mach. Learn.*, 43(1-2):97–119, 2001.

[3] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *SAC '03: Proceedings of the 2003 ACM symposium on Applied computing*, pages 784–788, New York, NY, USA, 2003. ACM Press.

[4] S. Dumais and H. Chen. Hierarchical classification of web content. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263, New York, NY, USA, 2000. ACM Press.

[5] C. Fox. A stop list for general text. *SIGIR Forum*, 24(1-2):19–21, r 90.

[6] L. Getoor, E. Segal, B. Taskar, and D. Koller. Probabilistic models of text and link structure for hypertext classification. In *IJCAI Workshop on "Text Learning: Beyond Supervision"*, Seattle, WA, August 2001.

[7] T. Joachims. Making large-scale support vector machine learning practical. pages 169–184, 1999.

[8] H.-J. Oh, S. H. Myaeng, and M.-H. Lee. A practical hypertext catergorization method using links and incrementally available class information. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 264–271, New York, NY, USA, 2000. ACM Press.

[9] M. F. Porter. An algorithm for suffix stripping. pages 313–316, 1997.

[10] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[11] R. E. Schapire and Y. Singer. BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[12] F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, 2002.

[13] A. Sun, E.-P. Lim, and W.-K. Ng. Web classification using support vector machine. In *WIDM '02: Proceedings of the 4th international workshop on Web information and data management*, pages 96–99, New York, NY, USA, 2002. ACM Press.

[14] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[15] W. J. Wilbur and K. Sirotkin. The automatic identification of stop words. *Journal of information science*, 18(1):45–55, 1992.

[16] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[17] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *J. Intell. Inf. Syst.*, 18(2-3):219–241, 2002.

0-7695-2841-4/07 $25.00 © 2007 IEEE