J.T. Yao, V.V. Raghvan, G.Y. Wang (Eds.): WSS'04, pp. 137-141, 2004.

Experiments in Web Page Classification for Semantic Web

Asad Satti, Nick Cercone, Vlado Kešelj
Faculty of Computer Science, Dalhousie University
E-mail: {rashid,nick,vlado}@cs.dal.ca

Abstract

We address the problem of web page classification within the framework of automatic annotation for Semantic Web. The performance of several classification algorithms is explored on the Four Universities dataset using page text and link information, with a limited-size feature set. Several well-known classification algorithms are evaluated on the task of web-page classification using the text of web pages. When compared to the methods that use link information, the text-based method shows surprisingly good performance, even with a feature set of limited size.

1 Introduction

The Semantic Web (SW) research area is concerned with transforming information available on the World Wide Web into knowledge by adding semantic structure to it. Embedded knowledge will help achieve efficient information retrieval, web-based question-answering, and intelligent agent-based applications. There are numerous and obvious benefits of having the web content enriched with semantic annotation; however, there are as many difficult challenges that need to be solved in order to pave the way for SW. One of them is defining conventions and ontologies for SW annotation, and another one is involved with the question of who will do the annotation: the web-page owner or an automatic tool. We advocate the use of automatic semantic annotation tools. There are several strong arguments why this is a preferred option: the first one is that the annotation requires significant and hardly justifiable effort on the part of the web-page owner so most of the owners will avoid it, and the second one is that an automatic annotation method will produce more uniform and consistent annotation.

Since the web encompasses many different domains it would be very difficult to try to annotate the whole web at once. We suggest breaking up the annotation of the entire web into the annotation of subsets representing different domains. After some evolution of the separate domain annotations, the problem of annotating the whole web can be tackled more easily. Because of this, the primary task in automatic annotation for SW is classification of web objects, typically pages, into domains. These domains are defined by *ontologies*. Operationally ontology can be described as the conceptual hierarchical organization/classification of concepts (classes). Since the web content and structure for a domain is composed by human beings under the implicit influence of their domain ontology e.g., yahoo directory, we believe that using domain ontology is the right direction. In any particular domain, concepts at the upper levels in the ontology hierarchy are more generic ones and many people agree on that with slightly varying terminology, but lower level concepts might be quite different. It is difficult to ensure that users annotate their web pages without bias.

To first step is to define the domain ontology using domain concepts (classes) and their hierarchical organization, their attributes, relations between the instances of two classes in the ontology, and inferences, if there are any. Semantic annotation of the web pages starts with the classification of web pages into ontology classes followed by extraction of attributes, and extraction of ontological relations between pages. For the World Wide Web and, in general, for any system with a large and growing number of entities classification becomes necessary for better understanding of the system.

2 Related Work

2.1 Classifiers

For our experiments we use the classifiers implemented in the WEKA package [8], which is an open source Java package containing various learning algorithms for classification, clustering, and association.

We apply the following classifiers: IBk, Naïve Bayes, J48, RIPPER, and PART. IBk is an implementation of the k-Nearest Neighbour (k-NN) method and finds the k most similar documents to the test document [1]. It then either assigns the same class to the test document that labels most of these k documents or class with maximum score after

weighted count of classes. For our experiment the value of k is one. The Naïve Bayes method is based on the Bayes' rule [1]. J48 is a WEKA version of the well-known C4.5 decision tree algorithm. RIPPER (Repeated Incremental Pruning to Produce Error Reduction) [2] is a propositional rule learner which is an optimized version of IREP [6]. PART [4] is also a rule learner that combined RIPPER and C4.5 algorithms, with pruning confidence threshold being 0.25 and minimum number of instances per leaf is 2.

Rule learning systems often scale relatively poorly with the sample size [2]. RIPPER rule learner was an effort to achieve efficiency for large noisy datasets and competitive generalization performance [2]. The core method REP (reduced error pruning) used for pruning decision trees can also be used for rule set pruning. It partitions the training data into a growing set and pruning set. First, a rule set is formed that overfits the growing set, then rule set is pruned using a pruning operator, which produces the greatest reduction of error on the pruning set from a set of operators. Pruning is terminated when an increase in the error on the pruning set is observed. REP for rules usually improves generalization performance on noisy data but it is computationally expensive. IREP (incremental reduced error pruning) was proposed by [6]. In this method, instead of growing the whole rule set first and then pruning it, each rule is pruned right after it is generated until the point when accuracy of the rule starts decreasing on the pruning set. Then pruned rule is added to the rule set and all positive and negative examples are removed from the training set (i.e., growing and pruning sets). When accuracy of a pruned rule drops below that of the empty rule then that rule is ignored and learned rule set is returned. RIPPER is realized after making three modifications to IREP namely, rule value metric, stopping condition, and rule set optimization in order to closely approximate reduced error pruning [2].

PART combines RIPPER and C4.5, the two rule learning paradigms [4]. Both RIPPER and C4.5 perform global optimization on the set of initially induced rules. Former does so to increase accuracy and later does to reduce the large rule set size. PART induces accurate and compact rule sets without using global optimization. Like RIPPER, it uses a divide-and-conquer strategy for building a rule i.e., it removes instances covered by the rule and continues recursively to create the rules for the remaining instance until no more instances are left [4]. In order to create each rule, a pruned decision tree is built from the current set of instances, then the leaf with the largest coverage is added to the rule set, and the tree is discarded. Algorithm to build pruned decision tree can be found in [4]. The use of pruned trees avoids the over-pruning problem in RIPPER.

2.2 Web Data Mining

The Four Universities dataset [7] consists of 8,282 web pages manually classified in to several classes. The pages are harvested from four universities, and they are kept in separate directories [3]. It was hypothesized in [5] that structural and link information on the web can make the classification of hypertext pages easier and more reliable instead of using just the page text.

3 Method

Since the dataset consists of web pages from four universities, we trained the classifier on three of the universities and tested it on the fourth university subset. This way, we avoid bias in evaluation results due to idiosyncrasies of all pages within one university. The classification evaluation is performed using 4-fold cross-validation [8], i.e., each time the training is performed on a different set of three university sets and the testing is performed on the remaining university set. Additionally, the miscellaneous data set is always used only in training.

A feature set to represent web pages is collected from the web page text after removing the HTML tags. The supporting features, such as link structure analysis, text around in-links and out-links, are not used. The number of features is restricted to the top 100 features, which are selected after ranking them with the information gain feature selection measure.

We test performance of several well-established classification methods. Namely, Naïve Bayes, J48 (pruned), RIPPER, k-Nearest Neighbour (k-NN), and PART have been evaluated to see which one achieves the best performance for the web page classification task.

To observe the class performance, the following standard evaluation measures are used:

- True Positive Rate (TP Rate), or Recall,
- False Positive Rate (FP Rate),
- Precision, and
- F-Measure.

The measures are defined by the following formulas:

$$\text{TP Rate} = \frac{\text{Correctly classified positives}}{\text{Total positives}}$$

$$\text{FP Rate} = \frac{\text{Incorrectly classified negatives}}{\text{Total negatives}}$$

$$\text{Precision} = \frac{\text{Correctly classified positives}}{\text{Total predicted positives}}$$

$$\text{F-Measure} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}$$

In addition to this, we use Percent of correctly classified and Percent of incorrectly classified instance to observe overall performance.

4 Evaluation Results

4.1 Dataset

The Four Universities dataset [7] has 8,282 web pages manually classified into student (1641), faculty (1124), staff (137) , department (182), course (930), project (504), and other (3764). Each class of data set contains web pages from 4 universities Cornell (867), Texas (827), Washington (1205), Wisconsin (1263), and miscellaneous (i.e., various other universities) (4120). The web pages for each university and miscellaneous are kept separately [3].

4.2 Classification Experiments

The summary of our experimental results is shown in Figure 1. Rows in the figure correspond to classifiers and columns show the test sets (i.e., held-out university sets). Our objective is to be able to classify the web pages from any new computer science department website using the learned model from the data set. The Miscellaneous (Misc) set is included in the training set to capture the different idiosyncrasies from many universities to improve the prediction accuracy of learners. In the figure, columns corresponding to each university show the percentage of instances correctly/incorrectly classified using corresponding test sets during 4-fold cross-validation and then these results are averaged in the average column. An experiment was performed by training on the 4 universities training set and testing on the Misc set, the results are as good as or better as compared to 4-fold cross-validation This shows that better efficiency can be achieved even without using the Misc set.

These results show that the k-Nearest Neighbour (k-NN) classifier completely outperforms the other classifiers and the percentage of instances correctly classified by k-NN is the highest among others. As in [4], our results show that C4.5 and PART give comparable results followed by RIPPER and Naïve Bayes.

Figure 2 shows the confusion matrix and information retrieval performance of Naïve Bayes. These values are calculated by averaging over the intermediate information retrieval results and confusion matrices, generated during the 4-fold cross-validation. It is interesting to note that one of the larger classes namely 'other' achieved more than 80% precision and another larger class 'student' achieved more than 80% recall. Since most of the instances of 'other' class are incorrectly classified into other classes, this accounts for the lower precision of the other classes. Many of instances

		TEST SETS					
		Cornell	Texas	Washington	Wisconsin	Average over 4 sets	Misc
	Total Instances	577	618	975	1060	3230	3200
CLASSIFIERS	Naïve Bayes % Correct	0.279	0.262	0.195	0.28	0.251	0.495
	Naïve Bayes %Incorrect	0.721	0.738	0.805	0.72	0.749	0.505
	J48 (C4.5) % Correct	0.856	0.853	0.927	0.865	0.88	0.883
	J48 (C4.5) % Incorrect	0.144	0.147	0.073	0.135	0.12	0.117
	PART % Correct	0.896	0.888	0.938	0.892	0.906	0.918
	PART % Incorrect	0.104	0.112	0.062	0.108	0.094	0.082
	RIPPER % Correct	0.851	0.79	0.912	0.799	0.841	0.787
	RIPPER % Incorrect	0.149	0.21	0.088	0.201	0.159	0.213
	IBk (k-NN) % Correct	0.998	0.99	0.997	0.996	0.996	0.995
	IBk (k-NN) % Incorrect	0.002	0.01	0.003	0.004	0.004	0.005

Figure 1. Percent correctly / incorrectly classified instances by different classifiers

of the other classes are incorrectly classified into the 'student,' this caused the lower recall of the other classes.

Figure 3 shows the results of C4.5 classifier. Lower precision of class 'department' is due to the class 'other.' Lower precision and recall of the 'staff' class is due to mix-up of its instance with 'student' class.

For the PART rule learner, shown in Figure 4, lower recall of class 'staff' is caused by its mix-up with the 'faculty,' 'student,' and 'other' class. RIPPER classifier also makes similar type of mistakes, as shown in Figure 5. Finally, k-NN results are shown in figure 6 and it gives very promising results.

These results show that on a scale of a web site, such as a university web site, we can rely on some classical, text-based methods for classification for purposes of SW annotation. In particular, the kNN algorithm demonstrates a very high performance.

5 Conclusion and Future Work

Web page classification is one of the core elements of our SW annotation system which is based on the automatic extraction of conceptual knowledge from web contents based on the given ontology. We also evaluate classifiers by automatically extracting significant terms from the web pages and using them as a feature vector. Results show that k-NN gives very good classification accuracy.

Our goal is to build ontology-driven information extraction methods based on the combination of pattern matching, machine learning, and natural language processing techniques. We will try to design an adaptive information extraction module which could adapt to the changing nature of the web. Finally annotation of the web will be done with the extracted knowledge, followed by an evaluation.

References

[1] S. Chakrabarti. *Mining the Web Discovering Knowledge from Hypertext Data*. Morgan-Kaufmann Publishers, 2003.

CLASSIFIER 1: Naive Bayes

Confusion Matrix								IR Performance Results				
a	b	c	d	e	f	g	<-- classified as	TP Rate	FP Rate	Precision	Recall	F-Measure
123	5	2	11	2	1	41	a = course	0.66475	0.10575	0.29225	0.6648	0.3995
0	2	0	0	0	0	0	b = department	1	0.031	0.0325	1	0.063
1	4	47	5	11	11	58	c = faculty	0.34025	0.0485	0.2485	0.3403	0.28625
354	97	129	238	167	96	1283	d = other	0.09575	0.0285	0.894	0.0958	0.172
0	1	3	3	23	9	32	e = project	0.32375	0.06275	0.1105	0.3238	0.16325
0	2	2	4	5	2	27	f = staff	0.02775	0.0455	0.01625	0.0278	0.0205
8	2	12	2	10	20	375	g = student	0.87375	0.5265	0.2145	0.8738	0.3415
							Average	0.475143	0.12121	0.258357	0.4751	0.20657143

Figure 2. True Positives Rate (TP Rate), False Positives Rate (FP Rate), Precision, and Recall of each class by using Naïve Bayes Classifier

CLASSIFIER 2: J48 (C4.5)

Confusion Matrix								IR Performance Results				
a	b	c	d	e	f	g	<-- classified as	TP Rate	FP Rate	Precision	Recall	F-Measure
157	1	0	22	2	0	3	a = course	0.8235	0.01075	0.83425	0.8235	0.8235
0	2	0	0	0	0	0	b = department	1	0.0035	0.225	1	0.3665
0	0	117	9	2	0	9	c = faculty	0.85325	0.01675	0.7105	0.8533	0.77125
36	11	26	2176	31	3	81	d = other	0.92225	0.1535	0.93825	0.9223	0.92925
0	0	1	17	46	0	7	e = project	0.64375	0.013	0.60425	0.6438	0.599
0	0	5	15	3	9	10	f = staff	0.1805	0.003	0.39275	0.1805	0.246
3	1	16	65	3	6	335	g = student	0.786	0.04	0.76475	0.786	0.7685
							Average	0.744179	0.03436	0.638536	0.7442	0.64342857

Figure 3. True Positives Rate (TP Rate), False Positives Rate (FP Rate), Precision, and Recall of each class by using C4.5 Classifier

CLASSIFIER 3: PART

Confusion Matrix								IR Performance Results				
a	b	c	d	e	f	g	<-- classified as	TP Rate	FP Rate	Precision	Recall	F-Measure
152	3	1	24	1	1	3	a = course	0.7985	0.0085	0.85775	0.7985	0.8215
0	2	0	0	0	0	0	b = department	1	0.0025	0.2915	1	0.45
1	1	122	6	3	1	3	c = faculty	0.88875	0.01575	0.72525	0.8888	0.79375
27	6	25	2215	34	5	52	d = other	0.93825	0.111	0.95425	0.9383	0.94575
0	0	0	18	49	1	3	e = project	0.68175	0.01375	0.53175	0.6818	0.59375
0	0	7	8	1	21	5	f = staff	0.465	0.00475	0.5495	0.465	0.50175
1	0	15	38	3	6	366	g = student	0.85075	0.02325	0.85875	0.8508	0.853
							Average	0.803286	0.02564	0.68125	0.8033	0.7085

Figure 4. True Positives Rate (TP Rate), False Positives Rate (FP Rate), Precision, and Recall of each class by using PART Classifier

CLASSIFIER 4: JRIP (RIPPER)

Confusion Matrix								IR Performance Results				
a	b	c	d	e	f	g	<-- classified as	TP Rate	FP Rate	Precision	Recall	F-Measure
166	0	1	18	0	0	0	a = course	0.87	0.023	0.70575	0.87	0.774
0	2	0	0	0	0	0	b = department	1	0.0015	0.4165	1	0.5835
0	0	104	23	2	0	8	c = faculty	0.751	0.01225	0.7425	0.751	0.74375
76	7	21	2140	37	0	83	d = other	0.913	0.281	0.89375	0.913	0.90075
0	0	1	39	28	0	3	e = project	0.40325	0.014	0.47175	0.4033	0.41875
0	0	2	19	1	8	12	f = staff	0.1735	0.00025	0.6875	0.1735	0.26875
3	0	11	140	7	1	267	g = student	0.63625	0.037	0.74975	0.6363	0.66
							Average	0.678143	0.05271	0.666786	0.6781	0.62135714

Figure 5. True Positives Rate (TP Rate), False Positives Rate (FP Rate), Precision, and Recall of each class by using RIPPER Classifier

CLASSIFIER 5: IBk (k-NN)

Confusion Matrix								IR Performance Results				
a	b	c	d	e	f	g	<-- classified as	TP Rate	FP Rate	Precision	Recall	F-Measure
185	0	0	0	0	0	0	a = course	1	0	1	1	1
0	2	0	0	0	0	0	b = department	1	0.0005	0.75	1	0.8335
0	0	137	0	0	0	0	c = faculty	1	0.00125	0.9715	1	0.98575
0	1	4	2359	0	0	0	d = other	0.99775	0.01075	0.9955	0.9978	0.99675
0	0	0	0	71	0	0	e = project	1	0	1	1	1
0	0	0	0	0	42	0	f = staff	1	0	1	1	1
0	0	0	9	0	0	420	g = student	0.9805	0	1	0.9805	0.99
							Average	0.996893	0.00179	0.959571	0.9969	0.97228571

Figure 6. True Positives Rate (TP Rate), False Positives Rate (FP Rate), Precision, and Recall of each class by using k-NN Classifier

[2] W. Cohen. Fast effective rule induction. In *Machine Learning: Proceedings of the Twelfth International Conference (ML95)*, 1995.

[3] M. DiPasquo, D. Freitag, D. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to construct knowledge base from the world wide web. *Artificial Intelligence*, 118:69–113, June 2000.

[4] E. Frank and I. Witten. Generating accurate rule sets without global optimization. In *Machine Learning: Proceedings of the Fifteenth International Conference*, San Francisco, CA, 1998. Morgan Kaufmann Publishers.

[5] J. Furnkranz. Using links for classifying web-pages. Technical Report OEFAI-TR-98-29, Austrian Research Institute for Artificial Intelligence, Wien, Austria, 1998.

[6] J. Furnkranz and G. Widmer. Incremental reduced error pruning. In *Machine Learning: Proceedings of the Eleventh Annual Conference*. Morgan Kaufmann Publishers, 1994.

[7] A. McCallum. The 4 universities data set. Retrieved Dec 3, 2003, *http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/*.

[8] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.