Table 3
The Co-Training algorithm

Given: a set LE of labeled training examples a set UE of unlabeled examples Create a pool UE' of examples by choosing u examples at random from UE. Loop until no examples are left in UE: Use LE to estimate the parameter set θ_1 of *Classifier1*. Use LE to estimate the parameter set θ_2 of *Classifier2*. Allow *Classifier1* with θ_1 to label p positive and n negative examples from UE'. Allow *Classifier2* with θ_2 to label p positive and n negative examples from UE'. Add these self-labeled examples to LE. Randomly choose $2p + 2n$ examples from UE to replenish UE'.

4.2.1. *Data set and experimental setting*

We collected the data set starting with four Web pages: a Japanese Web page,[3] two Thai Web pages,[4] and an English web page[5] From each of these four pages, a Web robot was used to recursively follow the links within the page until it retrieved 450 pages. Therefore, we had approximately 900 Thai pages as Thai pages might link to ones which were in English or other languages. We also had approximately 450 Japanese and 450 English pages. All of these pages were divided into three sets, denoted as A, B and C, each of which contained 600 pages (about 300 Thai, 150 Japanese and 150 English pages). Note that HTML mark-up tags were removed before the training and testing process. We used 3-fold cross validation in all experiments below for averaging the results. The settings for the classifiers were as follows.

(1) For ICT, we ran the algorithm without the consistency checking process. No labeled data was given to ICT. The initial θ_{10} was set to 0.7.
(2) For Co-Training, the values of the parameters of the classifier (in Table 3) were set in a similar way as in [1]. As Co-Training requires a small set of correctly pre-classified training data, we gave the algorithm with 18 hand–labeled pages. In our experiment, we set the values of $|UE|$, p, n and u to 1182, 3, 3 and 115, respectively.
(3) For EM, we supplied the algorithm with 18 initial labeled data and 1182 unlabeled data.
(4) For the supervised naive Bayes classifier, we gave the algorithm 1200 initial labeled data.

4.2.2. *Experimental results*

The results are shown in Table 4. In the table, "Co-Training(Bayes)" and "Co-Training(Word)" were the results of naive Bayes and word segmentation classifiers of Co-Training, respectively. "ICT(Bayes)" and "ICT(Word)" were for naive Bayes and word segmentation classifiers of ICT. As shown in the table, ICT(Word) gave the best performance according to F_1-measure, which was comparable to S-Bayes. The performance of ICT(Bayes) was higher than S-Word. Both classifiers of Co-Training had lower performance, compared to the other classifiers.

Compared to supervised learning classifiers, the performance of ICT was comparable to that of S-Bayes and quite better than that of S-Word. The results demonstrate that our system can effectively use unlabeled examples and the two modules succeed in training each other. From the experiments, we

[3] http://www.yahoo.co.jp.
[4] http://www.sanook.com, http://www.pantip.com.
[5] http://www.javasoft.com.

Table 4
The precision (%), recall (%) and F_1-measure of the classifiers for the problem of Thai/non-Thai page classification

Classifier	P (%)	R (%)	F_1
ICT(Word)	100.00	99.00	99.50
S-Bayes	100.00	99.00	99.50
ICT(Bayes)	100.00	98.78	99.39
S-Word	99.08	99.61	99.34
Co-Training(Bayes)	100.00	98.67	99.33
EM	100.00	98.56	99.28
Co-Training(Word)	100.00	98.45	99.22

found that ICT ran much faster than Co-Training and EM because Co-Training and EM used incremental labeling style during the training process which gradually added a small number of labeled data in each round. The learning process of ICT took 14.5 second, whereas Co-Training and EM took 51.5 and 37.2 second, respectively. Note that all of the algorithms were implemented using JAVA programming language and were run on Windows platform.

4.3. *The results of web page classification on the webKb data set*

The WebKb data set contains many Web pages related to the university domain. It was obtained via ftp from Carnegie Mellon University [11]. The data set consisted of 981 Web pages collected from the computer science department Web sites at four universities: Cornell, University of Washington, University of Wisconsin, and University of Texas. These Web pages were hand-labeled into four categories, which were course homepages, faculty homepages, project homepages and student homepages.

In this data set, some categories were actually closely related and this made the classification more difficult. A course home page gives information about the subject such as the course outline, the class schedule, reference books, etc. A faculty homepage is an instructor homepage, which gives information about the instructor's research, teaching course, etc. A project homepage is actually a research homepage. A student homepage is a personal homepage of a student in one of the universities.

4.3.1. *Data set and experimental setting*

We had 220 course Web pages, 147 faculty Web pages, 81 project Web pages and 533 student Web pages. Each sample was filtered to remove words that gave no significance in predicting the class of the document. Words to be eliminated were auxiliary verbs, prepositions, pronouns, possessive pronouns, phone numbers, digit sequences, dates and special characters. Then, the word stemming process was applied to each sample by using Porter algorithm [10] in order to remove all suffixes and search for similar words based on the root word. Finally, we extracted all headings appearing in each Web page to be the features of the heading-based classifier. Therefore, each Web page could be viewed as the set of words appearing in the page's content and the set of words appearing in all headings. The first classifier used the words appearing on the headings as the feature set, whereas the second classifier used words appearing in the content of the Web page as the feature set. The combined classifier predicted the class of examples based on the output from the heading-based and content-based classifiers as shown in Eq. 16.

$$Pr(l_j \mid d_i) = Pr(l_j \mid x_1)Pr(l_j \mid x_2) \tag{15}$$

where x_1 and x_2 are the heading and the content feature sets of document d_i. The settings for the classifiers were as follows.

(1) For ICT, we randomly selected 30% of all samples from each category to be initial labeled data. The training set (unlabeled data) consisted of 30% of all samples, and 40% of all samples were used as a test set.
(2) For Co-Training and EM, we used the same set of initial labeled data as ICT. The unlabeled data and the test set were also the same. The parameter p and n of Co-Training were set to 1 and 3, respectively.
(3) For the supervised naive Bayes classifier, we supplied the algorithm with 60% of the examples as labeled data and 40% of all samples were used as a test set.

4.3.2. Experimental results

The experiments were conducted using 5-fold cross-validation in order to give each Web page a chance to be trained and tested equally. After the training process was finished, we evaluated classifiers based on three feature sets, which were the heading feature set, the content feature set and the combined feature set (heading+content). After the learning process of every algorithms were accomplished, the performances of the algorithms were evaluated based on the feature sets.

Figures 3–5 show the results of classifiers using the heading feature set, the content feature set and the combined feature set, respectively. In the figures, "ICT" stands for the Iterative Cross-Training algorithm, "S-Bayes" stands for the supervised naive Bayes classifier. Co-Training and EM stands for the Co-Training and Expectation-Maximization algorithm, respectively. Considering the performance measured based on F_1 of classifiers using heading features (Fig. 3), we found that ICT got 80.73% on course homepages which was equal to Co-Training but higher than EM. This fact is also true for the student homepages, ICT got 92.45%, whereas Co-Training and EM got only 85.13% and 89.20%. For the project homepages, ICT obtained 50.06%, where as Co-Training and EM obtained 37.84% and 51.26%. Nevertheless, ICT gave the competitive performance compared to S-Bayes. S-Bayes got 81.13%, 50.00%, 55.56% and 91.16% measured on course homepages, faculty homepages, project homepages and student homepages, respectively.

For the content feature set as shown in Fig. 4, the F_1 score of ICT was 84.47% and was higher than those of Co-Training and EM on student homepages which were 66.27% and 77.05%, respectively. Nevertheless, the performance measured on other categories are less than those of Co-Training, EM and S-Bayes.

Figure 5 shows the performance measured on classifiers using the combined feature set (heading + content). The performance of ICT was higher than those of Co-Training and EM on course homepages and student homepages.

Note that we applied the micro average to measure the overall performance of each classifier. The micro average is normally used when the number of test data in each category are different. The average performance of all classifiers are given both in Fig. 6 and Table 5. Considering the average of F_1-measure, we found that the heading-based classifier of ICT obtained 78.25% correctness, which was higher than those of Co-Training and EM. The content-based classifier of ICT obtained 71.76% correctness, while Co-training and EM got 66.14% and 70.70%, respectively. Nevertheless, the performance of classifiers using ICT was a bit less than those using S-Bayes. This is because ICT employed only 50% of the labeled data used by S-Bayes. The performances of classifiers using both feature sets of ICT were also higher than EM and Co-Training.

We found that the training time of ICT was much less than Co-Training and EM. With ICT, it took about 3 minutes for the algorithm to converge, whereas with Co-Training and EM, it took more than 20 minutes to converge (all of the algorithms were implemented using JAVA language on Windows platform).

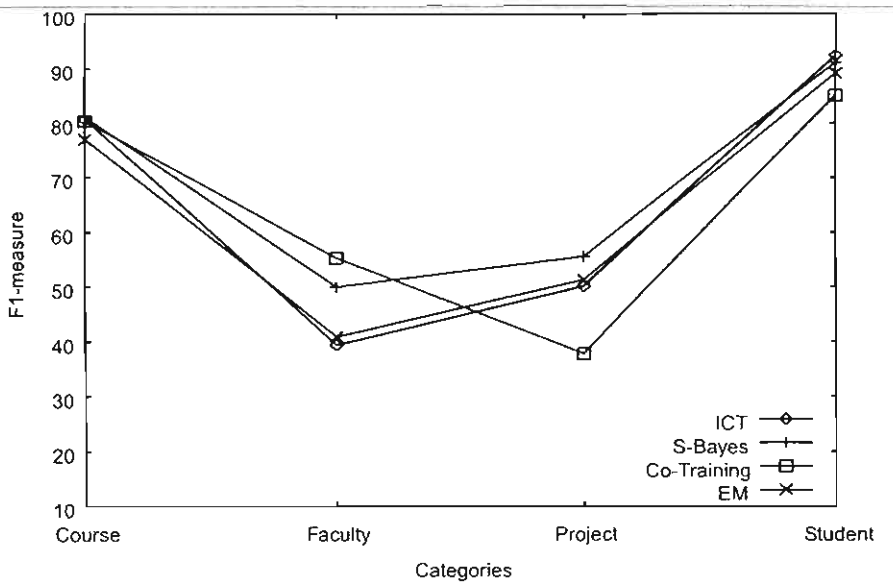


Fig. 3. The performance of classifiers using the heading feature set on the WebKb data set.

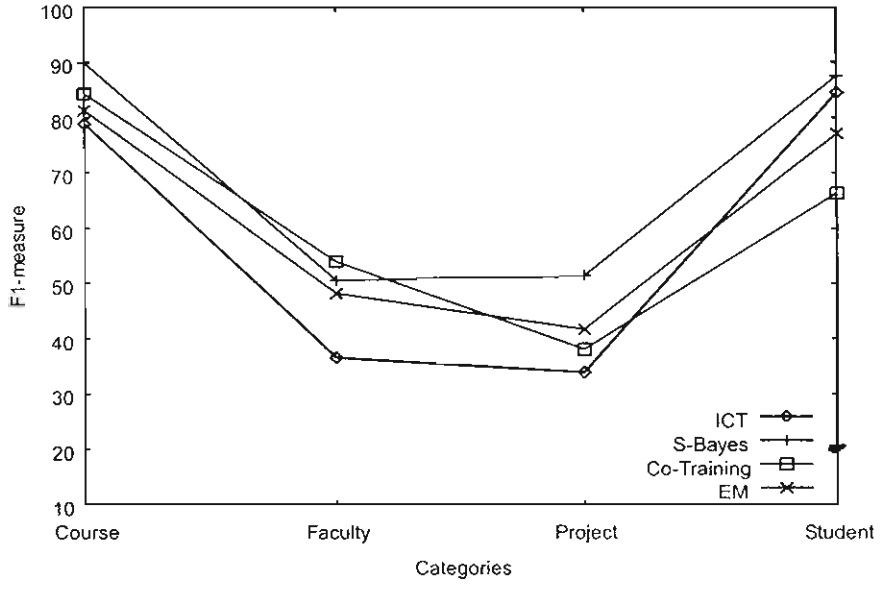


Fig. 4. The performance of classifiers using the content feature set on the WebKb data set.

4.4. *The results of web page classification on the webClass data set*

The WebClass data set was obtained from a machine learning research group in Italy [12]. It consists of 92 Web pages corresponding to four categories, which are astronomy, jazz, auto and motorcycle. Each category has 48 pages. The first two categories are semantically distant, whereas auto and motorcycle categories are both concerning about vehicles and are closely related.

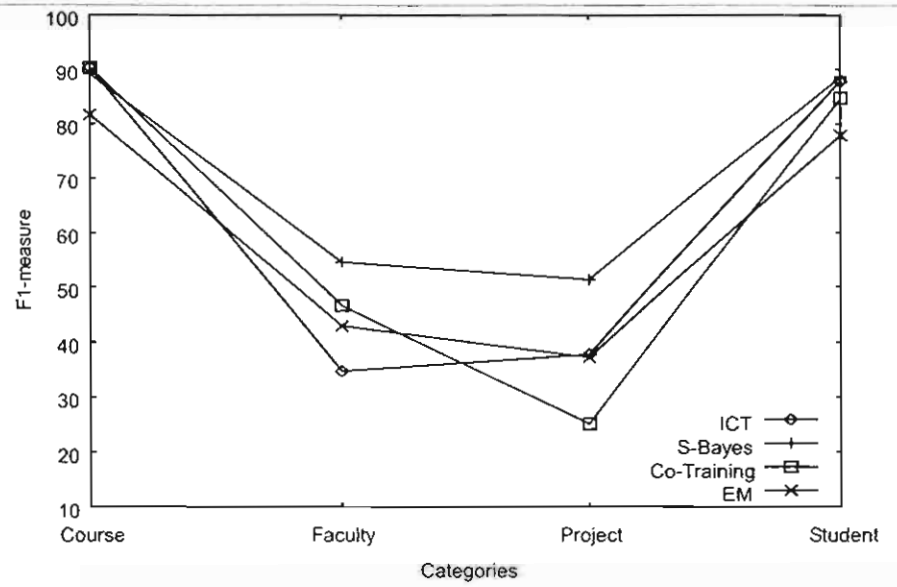


Fig. 5. The performance of classifiers using the combined feature set on the WebKb data set.

Table 5
The average performance of classifiers on the WebKb data set

Classifier	Heading			Content			Heading+Content		
	P (%)	R (%)	F_1	P (%)	R (%)	F_1	P (%)	R (%)	F_1
ICT	71.85	94.39	78.25	67.23	86.73	71.76	73.39	88.27	76.22
S-Bayes	74.99	87.24	79.91	76.95	84.14	79.60	78.92	83.76	80.46
Co-Training	73.95	84.69	75.64	79.69	60.72	66.14	68.29	87.26	75.30
EM	68.62	91.64	75.98	76.78	74.28	70.70	74.87	78.50	70.08

4.4.1. Experimental setting

The preprocessing step was done in the same way as in the WebKb data set. The settings for classifiers were as follows.

(1) For ICT, Co-Training and EM, we selected 33% of all examples to be initial labeled data. The training set consisted of 33% and the remaining 34% was a test set.
(2) For the supervised naive Bayes classifier, we selected 66% of all examples to be labeled data. The test set was also 34% of all examples.

4.4.2. Experimental results

All experiments were conducted using 3-fold cross-validation. Figures 7–9 show the results of classifiers measured based on heading, content and the combined feature sets.

Considering performance measured based on heading features on classifiers as shown in Fig. 7, ICT got 96.55% on astro homepages which was higher than those of S-Bayes, Co-Training and EM which got 84.08%, 76.47% and 89.51%, respectively. For the motorcycle homepages, ICT got 84.62% which was higher than those of S-Bayes, Co-Training and EM which got 81.27%, 70.04% and 52.72%, respectively. Nevertheless, the performances of the other categories, auto and jazz, of ICT were less than those of S-Bayes, Co-Training and EM.

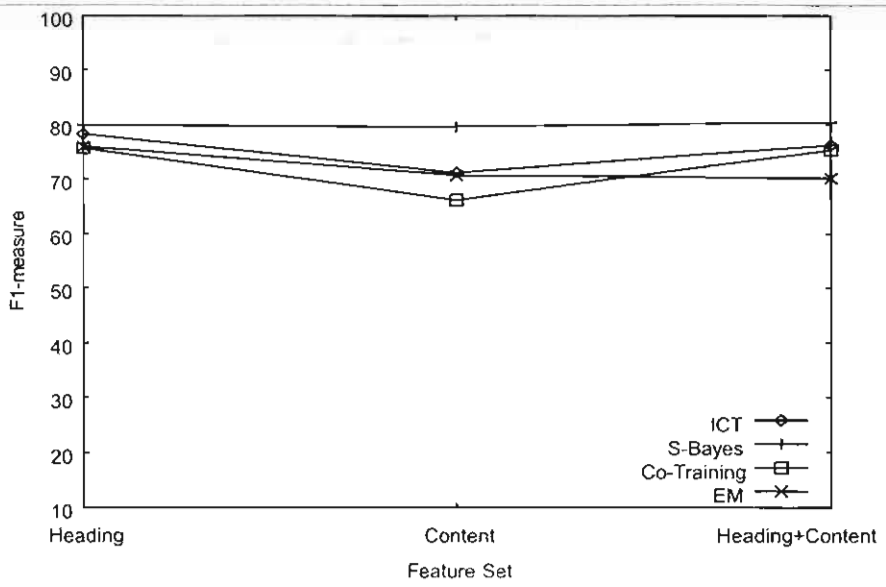


Fig. 6. The average performance of classifiers using different feature sets on the WebKb data set.

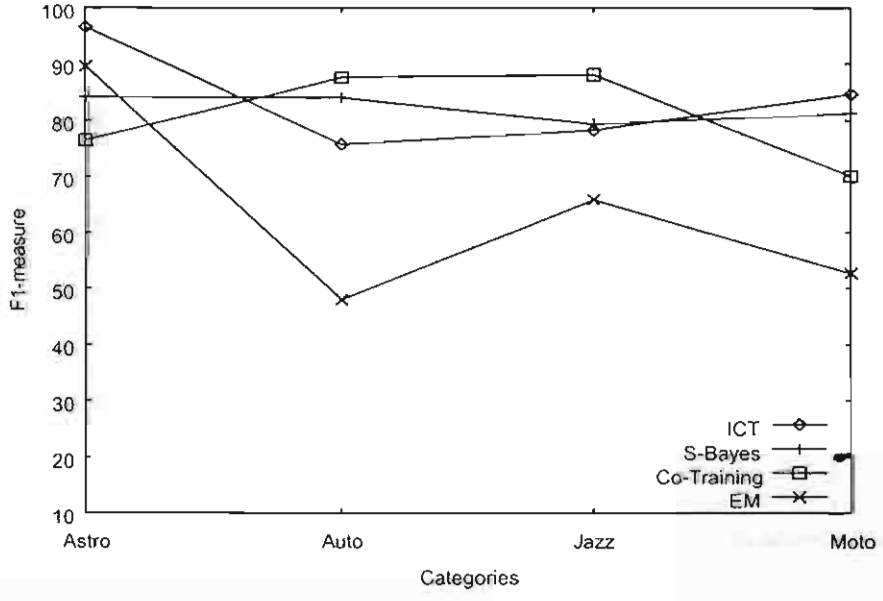


Fig. 7. The performance of classifiers using the heading feature set on the WebClass data set.

Figure 8 shows the performance of classifiers using the content feature set. We found that ICT got higher performance than other classifiers on auto, jazz and motocycle categories. The performance measured based on the combined feature sets is shown in Fig. 9. We found that ICT got higher performance on jazz and motocycle categories (100.00%, 96.55%), whereas S-Bayes, Co-Training and EM got 98.85%, 92.97%, 97.78% on the jazz category and 90.39%, 82.91%, 85.35% on the motorcycle category.

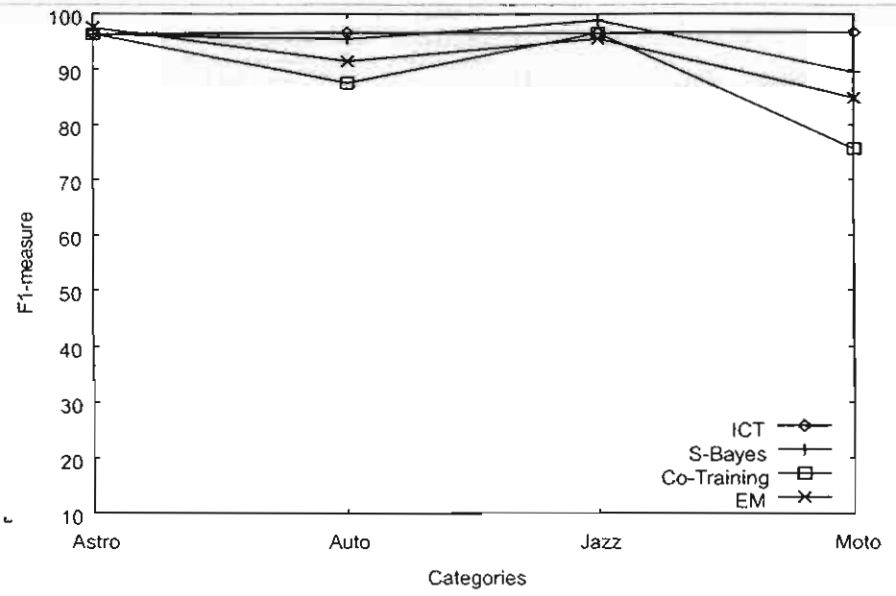


Fig. 8. The performance of classifiers using the content feature set on the WebClass data set.

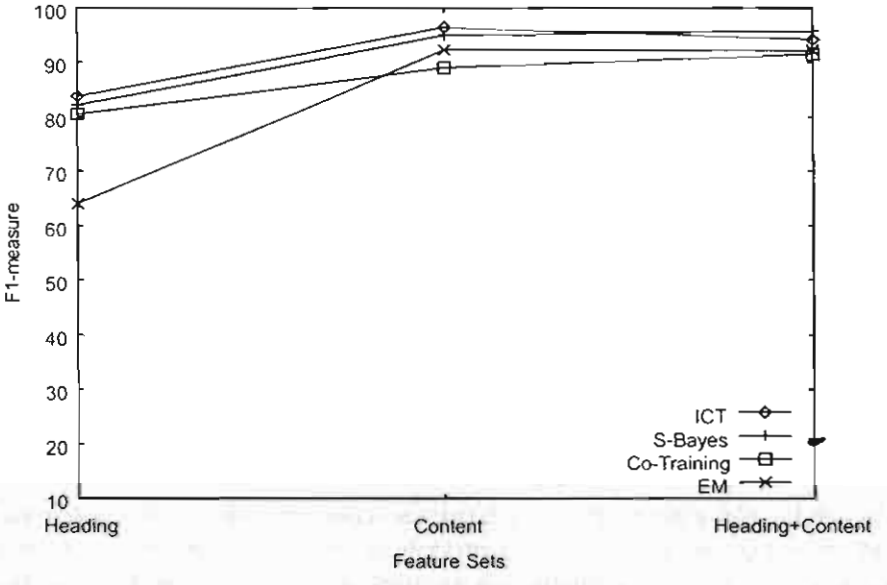


Fig. 9. The performance of classifiers using the combined feature set on the WebClass data set.

Considering the average performance measured by F_1 (as shown in Fig. 10 and Table 6), we found that the heading-based classifier of ICT obtained 83.78%, which was higher than those of Co-Training and EM.

The performance of classifiers using the combined feature set of ICT was higher than those using S-Bayes. However, the performance deficiency of the combined feature set (heading+content-based classifier) was less than those using S-Bayes. For the heading-based classifier, Co-Training and EM lost

Table 6
The average performance of classifiers on the WebClass data set

Classifier	Heading			Content			Heading+Content		
	P (%)	R (%)	F_1	P (%)	R (%)	F_1	P (%)	R (%)	F_1
ICT	86.47	85.72	83.78	95.00	98.22	96.49	92.45	96.43	94.18
S-Bayes	84.36	85.12	82.17	93.14	97.62	94.99	93.97	98.21	95.81
Co-Training	74.16	92.86	80.51	83.00	98.22	89.01	87.62	97.02	91.43
EM	68.11	69.64	64.00	87.60	98.81	92.31	87.31	99.40	92.21

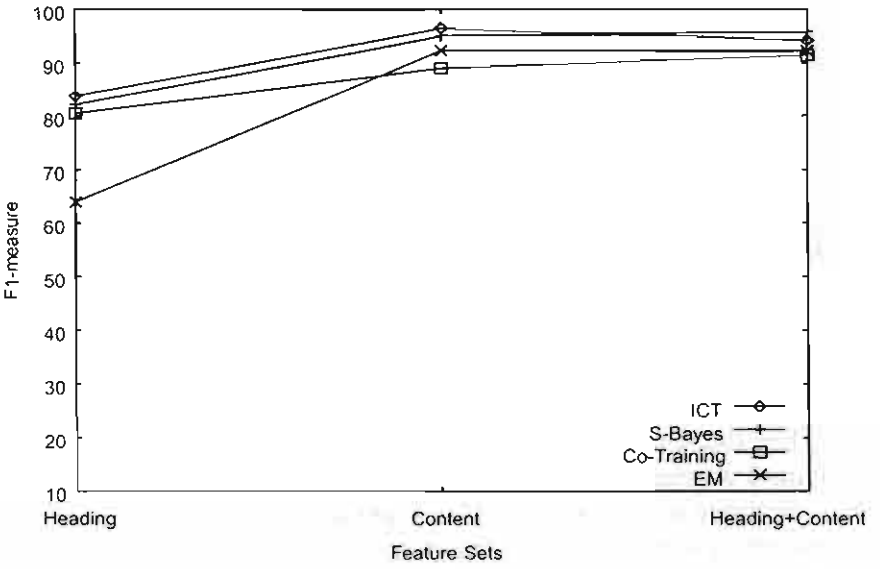


Fig. 10. The average performance of classifiers using different feature sets on the WebClass data set.

2.02% and 22.11%, respectively.

For the content-based classifiers, the classifier of ICT got higher performance than S-Bayes. Moreover, the classifiers of Co-Training and EM lost 6.30% and 2.81%, respectively. Note that the training time of ICT is much less than Co-Training and EM. With ICT, it took 1 minute for the algorithm to converge, whereas Co-Training took 5 minutes and EM took 2 minutes.

4.5. *Performance of ICT on noisy data*

In reality, there is a possibility that the initial labeled data are incorrectly labeled due to human error. Therefore, it is interesting to see how a learning algorithm is affected by this real world problem. Since ICT, Co-Training and EM are boosting-style learning algorithms, they need a small amount of initial labeled data. Therefore, we added noise to these labeled data and let the algorithms start the learning process.

The experiments were conducted on three data sets as in the previous subsections. Varying levels of random class noise, between 10% to 50% were added to the initial labeled data. The results are shown below.

As shown in Fig. 11, ICT was robust in the presence of noise as neither classifier's performance changed. The word segmentation and naive Bayes classifiers of ICT still preserved their performance at 99.50% and 99.39%, when noise was added up to 50%. This was because all unlabeled data were

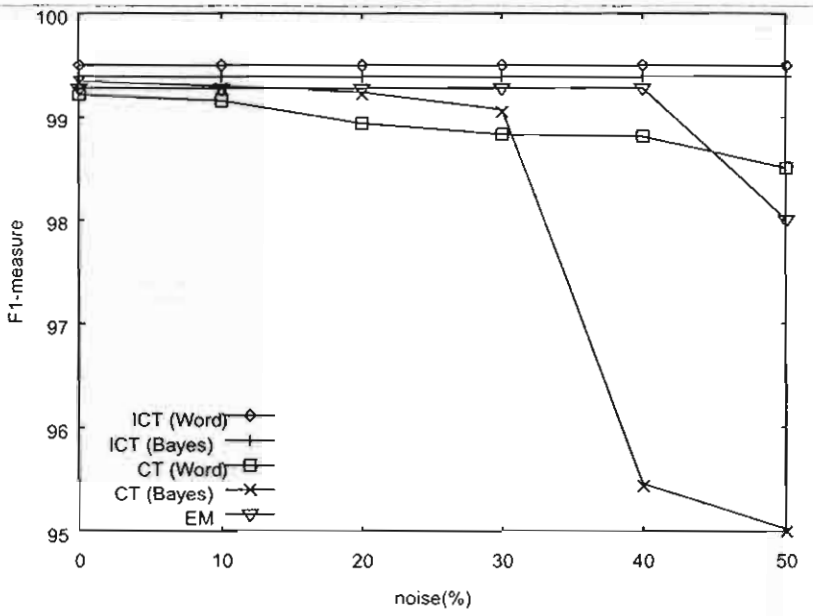


Fig. 11. The performance of classifiers at different levels of noise on the Thai/nonThai data set.

relabeled in every iterations by both classifiers. Both classifiers, CT (Word) and CT (Bayes), of Co-Training were sensitive to noise as their performances dropped considerably from 99.22% to 98.50%, and from 99.33% to 95.00%. The reason that Co-Training was more sensitive is that Co-Training used an incremental labeling style and thus the noisy initial labeled data had very high influence to each classifier of Co-Training. Since the classifiers learned from the incorrectly labeled data would very likely assign the new wrong labels incrementally to the unlabeled data. These new mislabeled data would be accumulated during the training process, which caused the performance degradation of Co-Training. The performance of the EM algorithm dropped slightly when noise was increased to 50%.

The graphs in Fig. 11 show the performances of all classifiers on the WebKb data set. We found that, when noise was added up to 50%, the heading-based and content-based classifiers of ICT lost about 10.85% and 12.70%, respectively. Both classifiers of Co-Training lost 14.58% and 16.17% of performance. Considering the EM algorithm, we found that the loss of performance due to noise labeled data is still acceptable. The heading-based classifier of EM lost 11.99% of correctness, which was comparable to ICT. The content-based classifier of EM lost 20.79%

The performances of all classifiers when noise was added on the WebClass data set are shown in Fig. 13. We found that both classifiers of ICT lost less performance than those of Co-Training. The performance loss of heading-based and content-based classifiers of ICT was 20.92% and 24.10%, whereas the heading-based and content-based classifiers of Co-Training lost 31.70% and 31.47%. For the EM algorithm, the heading-based and content-based classifiers lost 18.75% and 35%, respectively.

5. Conclusions

We have presented an algorithm that effectively uses unlabeled examples to estimate the parameters of the system for classifying Web pages. The method is based on two sub-classifiers that iteratively train

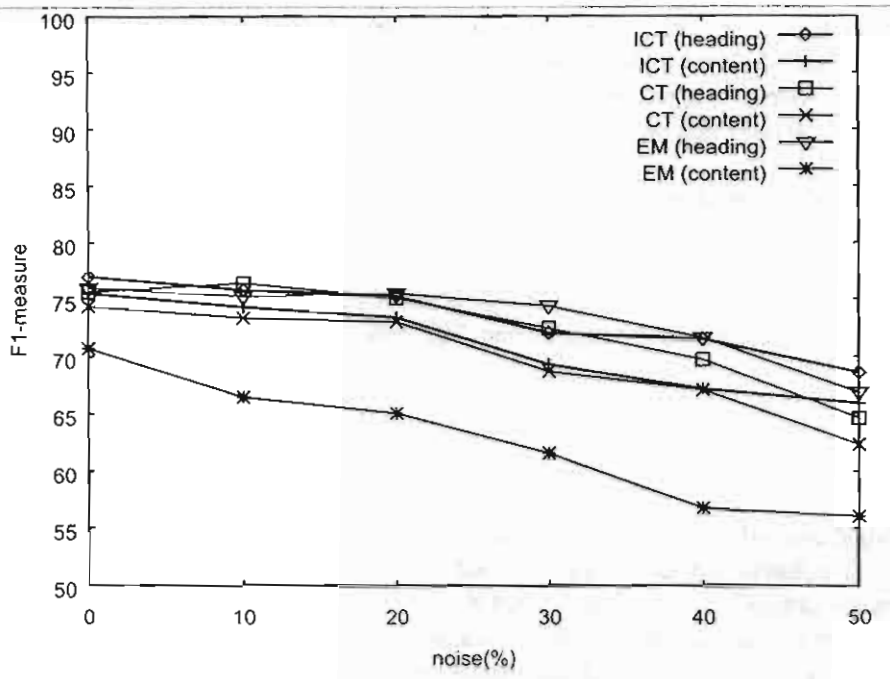


Fig. 12. The performance of classifiers at different levels of noise on the WebKb data set.

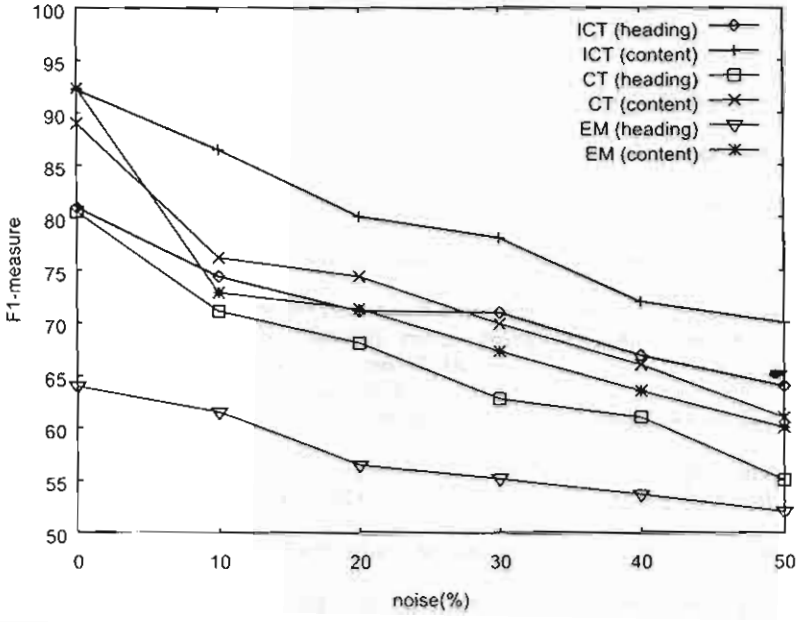


Fig. 13. The performance of classifiers at different levels of noise on the WebClass data set.

each other. Since ICT consists of two sub-classifiers, the algorithm has an ability to utilize different feature sets of the Web pages during the learning process to construct the most appropriate model for classifying unseen Web pages. With no or a small set of pre-labeled examples, our method gives high

precision and recall on classifying Web pages. The performance of our method is comparable with those of supervised ones, which demonstrates the successful use of unlabeled data of our algorithm. We have applied ICT to three classification problems. When we supply domain knowledge to the stronger classifier, ICT has an ability to boost the performance of the weaker classifier. Moreover, ICT's performance is still acceptable, when no domain knowledge is given.

ICT has an advantage in that it quickly converges since the labeling style is re-labeling mode, unlike Co-Training which uses an incremental labeling style. ICT is robust in the presence of noise, when we can provide domain knowledge to the algorithm. ICT is an easy to use algorithm, since we need not tune many parameters, unlike the Co-Training algorithm which requires the parameter tuning of p and n. In the case that no domain knowledge was available, ICT performance declines less than Co-Training and EM.

Acknowledgements

The authors would like to thank Prof. Luc De Raedt from the Machine Learning and Natural Language Processing Research Group, University of Freiburg, for his suggestions and valuable comments. This research has been supported by the Thailand Research Fund, and National Electronics and Computer Technology Center. The views and conclusions contained in this paper are those of the authors and the Thailand Research Fund or National Electronics and Computer Technology Center is not necessarily of the same opinion.

References

[1] A. Blum and T. Mitchell, *Combining labeled and unlabeled data with cotraining*, in Proc. Int. Conf. the 11th Annual Conference on Computational Learning Theory, 1998.
[2] A.P. Dempster, N.M. Laird and D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, *Royal Statistical Society* 39(1) (1997), 38.
[3] T. Joachims, *Text categorization with support vector machines: Learning with many relevant features*, in: Proc. Int. Conf. Machine Learning, 1998, pp. 137–142.
[4] A.R. Jones, A. McCallum, K. Nigam and E. Rilo, *Bootstrapping for text learning tasks*, in Proc. Int. Conf. Artificial Intelligence, 1999, pp. 52–63.
[5] A. McCallum and K. Nigam, *Employing em and pool-based active learning for text classification*, 1998, pp. 350–358.
[6] S. Meknavin, P. Charoenpornsawat and B. Kijsirikul, *Feature-based thai word segmentation*, in Proc. Int. Natural Language Processing Pacific Rim Symposium, 1997, pp. 41–48.
[7] T. Mitchell, Machine Learning, McGraw-Hill, 2 ed., 1979.
[8] K. Nigam, A. McCallum, S. Thrun and T. Mitchell, *Text classification from labeled and unlabeled documents using em*, 2000, pp. 103–134.
[9] J.M. Pierre, Practical issues for automated categorization of web sites, *In Proc. Int. Conf. Semantic Web*, 2000.
[10] M.F. Porter, An algorithm for suffix stripping, PhD thesis, available on-line at http://www.dcs.gla.ac.uk/Keith/ Preface.html., 1980.
[11] Web-Kb Project, http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo51/www/co-training/data/course-cotrain-data.tar.gz. Carnegie Mellon University, USA.
[12] WebClass Project, http://www.di.uniba.it/malerba/software/webclass/webclass.html, University of bari, Italy.
[13] C.J. van Rijsbergen, *Information Retrieval. Dept. of Computer Science*, University of Glasgow, 2 ed., 1979.
[14] Y. Yang and J. Pederson, *Feature selection in statistical learning of text categorization*, In Proc. Int. Conf. Machine Learning, 1997, pp. 412–420.

tapraid5/8c-int/8c-int/8c1203/8c0419-03a | franklim | S=4 | 9/22/03 | 7:18 | Art: 3049L | Input-bsu(bsu)

Iterative Cross-Training: An Algorithm for Learning from Unlabeled Web Pages

AQ: 1

Nuanwan Soonthornphisaj,* Boonserm Kijsirikul†
Machine Intelligence and Knowledge Discovery Laboratory, Department of Computer Engineering, Chulalongkorn University, Bangkok 10330, Thailand

The article presents a new learning method, called *iterative cross-training* (ICT), for classifying Web pages in three classification problems, i.e., (1) classification of Thai/non-Thai Web pages, (2) classification of course/non-course home pages, and (3) classification of university-related Web pages. Given domain knowledge or a small set of labeled data, our method combines two classifiers that are able to use effectively unlabeled examples to iteratively train each other. We compare ICT against the other learning methods: a supervised word segmentation classifier, a supervised naïve Bayes classifier, and a co–training-style classifier. The experimental results on three classification problems show that ICT gives better performance than those of the other classifiers. One of the advantages of ICT is that it needs only a small set of prelabeled data or no prelabeled data in the case that domain knowledge is available. © 2003 Wiley Periodicals, Inc.

AQ: 2

1. INTRODUCTION

Given prelabeled training data, supervised learning has been applied successfully to text classification.[1,3,4,7,8,10,18] However, one of the difficulties of using supervised learning is that the algorithm needs a large number of labeled examples to find the common properties of the class and use them to classify unseen data. Unfortunately, the text classification is a tedious job and a time-consuming task for humans to read and analyze the category of the pages. Although it is costly to construct hand-labeled data, in some domains it is easy to obtain unlabeled data such as data on the Internet. Thus, if we are able to use effectively the available unlabeled data, we will simplify the task of building text classifiers. Various methods have been proposed to use unlabeled data together with prelabeled data for text classification, such as *active learning with committee,*[11] text classification using EM,[15] and the *co-training* algorithm.[2]

AQ: 3

AQ: 4

*Author to whom all correspondence should be addressed: e-mail: nuanwan@mind.cp.eng.chula.ac.th.
†e-mail: boonserm@mind.cp.eng.chula.ac.th.

AQ: 20

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 18, 000–000 (2003)
 Published online in Wiley InterScience (www.interscience.wiley.com). • DOI 10.1002/int.10157

This article describes a new algorithm, called *iterative cross-training* (ICT), that effectively uses unlabeled data in the domain of Web page classification where unlabeled data is plentiful and easy to obtain. Our method combines two classifiers, which iteratively train each other. Given two sets of unlabeled data, each of which is for each classifier, the classifiers label the data for the other. The first classifier is given some knowledge about the domain and uses the knowledge to estimate labels of the examples for the second classifier. The second classifier has no domain knowledge and learns its model from examples labeled by the first, using the current model to label training data for the first. This training process is iteratively repeated. With good interaction between two classifiers, the performance of the whole system is increasingly improved. If we have no domain knowledge, we supply the algorithm with a small number of labeled examples. One of the advantages of our method is that because the method requires no labeled data or needs only a small number of data, it reduces human effort in labeling data and can be trained easily with a lot of unlabeled data.

Because the Internet becomes a part of our life, everyone could access any Web pages through a search engine. To provide the high impact to users and apply our learning algorithm to the real-world application, we intend to apply our learning method in the area of the Internet that focused on Web page classification. Therefore, we apply our algorithm to three classification problems: (1) the classification of Web pages based on the specific language (Web pages written in Thai language and non-Thai language); (2) the classification of Web pages into course and noncourse home pages, which was introduced by Blum and Mitchell[2]; and (3) the classification of the university-related home page, which was introduced by Craven et al.[5] To evaluate the effectiveness of our method, we also implement other classifiers to compare empirically with our method. The implementation is designed to explain or at least give some answers to questions: Is ICT that combines two classifiers an effective method? Does this kind of combination of two classifiers perform better than only one? Can the method successfully use unlabeled data? The other classifiers are (1) a supervised word segmentation classifier (S-Word), (2) a supervised naïve Bayes classifier (S-Bayes), (3) a co–training-style classifier (CoTraining). Among these classifiers, S-Bayes or S-Word is a single and supervised classifier. CoTraining and ICT are composed of two subclassifiers and are able to use unlabeled data.

The experimental results show that ICT successfully and efficiently classifies Web pages with high precision and recall. The overall performance, evaluated by F_1 measure of ICT is better than those of the other methods tested in our experiments. The better performance of ICT than those of supervised ones (e.g., S-Bayes) shows the successful use of unlabeled data. The results also show that the training technique of ICT also is an effective way because its performance is better than that of CoTraining, which uses a different training technique.

This study is organized as follows. Section 2 describes an overview of our system and gives the details of our classifiers. Section 3 describes other learning methods used in our comparison. Section 4 describes the experimental results. Discussion and related work are given in Section 5 and, finally, Section 6 concludes our work.

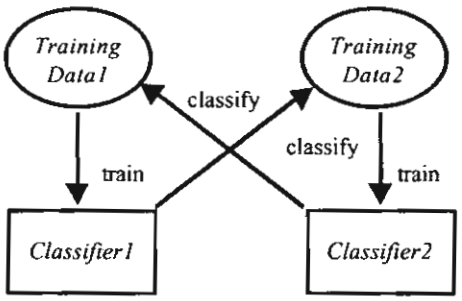


Figure 1. The architecture of ICT is shown. It is composed of two classifiers that use unlabeled data to iteratively train each other.

2. ICT

This section presents ICT. First, we describe the architecture of our learning system and then give the details of two classifiers used in the system. Figure 1 F1 shows our learning system, which determines how to classify Web pages. The system is composed of two classifiers: *Classifier1* and *Classifier2*. Given domain knowledge or a small set of prelabeled data, these two classifiers estimate their parameters from unlabeled data by receiving training from each other. Two training data sets, called *TrainingData1* and *TrainingData2* are duplicated from the unlabeled data provided by the user. Let θ_1 and θ_2 be sets of parameters of *Classifier1* and *Classifier2*, respectively. *TrainingData1* is used to train *Classifier1* to estimate its parameter set, and *TrainingData2* is used to estimate the parameter set of *Classifier2*. The algorithm for training the classifiers is shown in Table I. T1

The idea behind our algorithm is that if we can obtain reliable statistical information contained in *TrainingData2*, it should be useful in classifying *TrainingData1*. If the starting parameter set of *Classifier1* (θ_{10}) has the property that it produces more true positive than wrong positive examples and more true negative than wrong negative examples for *TrainingData2*, the statistical information in correctly classified examples will be obtained.

Using this information, *Classifier2* should correctly classify more examples in *TrainingData1* that have similar characteristics. If the newly labeled *TrainingData1* can produce θ_1 better than θ_{10}, more reliable parameters of the whole system should be obtained after each iteration.

In the algorithm, first, we initialize the parameter sets of *Classifier1* and *Classifier2*. This is done by training the classifiers with a small set of labeled examples if they are available. If no labeled example is provided to the system, the values of the parameters can be a predetermined or randomly chosen. When a classifier labels data, it can ask for the confirmation from the other classifier to make decisions about which class the example should be. If both classifiers agree with the same classifying result, that example will be labeled. The purpose of consistency checking is to produce more reliable labeled data, but the checking will slow down the learning process.

Table I. The training algorithm of ICT.

Given:
- Two sets *TrainingData1* and *TrainingData2* of unlabeled training examples

Initialize the parameter set of *Classifier1* to θ_{10}
$\theta_1 \leftarrow \theta_{10}$
Initialize the parameter set of *Classifier2* to θ_{20}
$\theta_2 \leftarrow \theta_{20}$
Loop until θ_1 does not change or the number of iterations exceeds a predefined value:
If *labeling_mode* = BATCH Then
- Use *Classifier1* with the current parameter set θ_1 to label all data in *TrainingData2* into positive examples and negative examples, and check consistency of the classification with *Classifier2* if necessary.

Else * *labeling_mode* = INCREMENTAL *\
- Use *Classifier1* with the current parameter set θ_1 to label the class for the most confident p-positive unlabeled examples and most confident n-negative unlabeled examples, and check consistency of the classification with *Classifier2* if necessary.

Train *Classifier2* by using labeled examples in *TrainingData2* to estimate the parameter set θ_2 of *Classifier2*.
If *labeling_mode* = BATCH, Then
- Use *Classifier2* with the current parameter set θ_2 to label all data in *TrainingData1* into positive examples and negative examples, and check consistency of the classification with *Classifier1* if necessary.

Else * *labeling_mode* = INCREMENTAL *\
- Use *Classifier2* with the current parameter set θ_2 to label the class for the most confident p-positive unlabeled examples and most confident n-negative unlabeled examples, and check consistency of the classification with *Classifier1* if necessary.

Train *Classifier1* by the labeled examples in *TrainingData1* to estimate the parameter set θ_1 of *Classifier1*.

As shown in Table I, the algorithm has two labeling modes: *batch labeling* and *incremental labeling*. The user must specify which labeling mode will be used in a particular problem. The difference between these two labeling modes is how the algorithm labels the data. In incremental mode, the algorithm will incrementally produce a small set of new labeled examples at each round, but in batch mode, the algorithm will label all examples and relabel them at each round. The batch-mode labeling tends to run fast, and the incremental-mode labeling tends to be more robust.

The following sections describe the details of the classifiers.

2.1. Subclassifiers in ICT for the Classification of Thai/Non-Thai Web Pages

In the problem of classification of Thai/Non-Thai Web pages, our goal was to classify Web pages into Thai and non-Thai pages. This problem is of interest because we want to build a Web robot that efficiently crawls the Web and retrieves only Thai pages for building a Thai search engine. In this problem, the first subclassifier *Classifier1* is given some knowledge about the domain in the form of a dictionary and uses the dictionary for helping in determining whether a page is

written in Thai or not. The algorithm used by *Classifier1* is the word segmentation algorithm, which will be described in the next section. The second subclassifier *Classifier2* is given no knowledge and uses the naïve Bayes classifier.

2.1.1. *Word Segmentation Classifier (Classifier1)*

One straightforward way to determine whether a Web page is in a specific language is to check the words in the page with a dictionary. If *many* words appear in the dictionary, it is likely that the page is in that language. We can not expect that all words in the page appear in the dictionary because the Web page usually contains names of persons, organizations, etc. not occurring in the dictionary and may contains words written in foreign languages. Therefore, it is necessary to determine how many words should be contained. This task is more difficult when it is considered in a language that has no word boundary delimiters such as Thai, Japanese, etc.[13]

Note that a string of Thai characters usually can be segmented in many possible ways because a word may be a substring of a longer word, and without a word delimiter, it is difficult to find which segmentation is correct. Next, we describe our method for word segmentation.

Given a Thai dictionary, a document d of n characters $(c_1, c_2, \ldots, c_n)$, the word segmentation classifier generates all possible segmentations and finds the best segmentation $(w_1, w_2, \ldots, w_m)$ that minimizes the cost function in Equation 1.

$$\underset{w_1,\ldots,w_m}{\operatorname{argmin}} \sum_{i=1}^{m} \operatorname{cost}(w_i) \tag{1}$$

where $\operatorname{cost}(w_i)$ is $\eta 1$ if w_i is a word in the dictionary and $\eta 2$ if w_i is a string not in the dictionary.

In the following experiments, $\eta 1$ and $\eta 2$ are set to 1 and 2, respectively. Because generating all possible segmentations and calculating their costs is very expensive, we use a dynamic programming technique to implement this calculation. Note that any sequence of characters, $c_i, \ldots, c_j$, found in the dictionary must be considered as a word and must not be grouped with nearby characters to form a long unknown string.

After the best segmentation is determined, the document is composed of (1) words that appeared in the dictionary and (2) unknown strings not found in the dictionary. A Thai Web page should be the page that contains many words and few unknown strings. We then define *WordRatio* of a page as:

$$\frac{\text{the number of characters in all words}}{\text{the number of all characters in the document}}$$

Given sets of positive and negative examples, the classifier finds the threshold of *WordRatio* that maximizes the number of correctly classified positive and negative examples. If *WordRatio* of a page is greater than the threshold, we will classify it as positive (Thai page). Otherwise, we will classify it as negative (non-Thai page).

For simplicity, let us use only the threshold of *WordRatio* as the parameter of the word segmentation classifier (θ_1).

Having only the threshold of *WordRatio* (θ_1) as the parameter, we can find θ_{10}, which produces more true positive and true negative examples for *TrainingData2*. As described previously, most of Thai pages should have a high value of *WordRatio*, whereas non-Thai pages should have a low value of *WordRatio*. If the numbers of Thai and non-Thai pages in *TrainingData2* are the same, it is easy to see that any value of θ_{10} will give more correctly classified pages than incorrectly classified pages (except for $\theta_{10} = 0.0$ or $\theta_{10} = 1.0$, which gives the same number of correctly and incorrectly classified pages). If the number of Thai pages is lower than the number of non-Thai pages, a high value of θ_{10} (e.g., 0.7, 0.8, and 0.9) will produce more correctly classified pages. This is the case that is likely to be encountered in the real world. A low value of θ_{10} is used when the number of Thai pages is larger than that of non-Thai pages. AQ: 5

A new θ_1 can be estimated after the naïve Bayes classifier (*Classifier2*) labels data in *TrainingData1*. Let *SP* be the smallest value of *WordRatios* from all labeled positive examples, and let *LN* be the largest value from all labeled negative examples. In case of $SP \geq LN$, the new θ_1 is estimated as

$$\theta_1 = \frac{SP + LN}{2} \tag{2}$$

Now, consider the case of $SP < LN$. Let $V_1 = SP$, $Vn = LN$, and $V_2, \ldots, V_{n-1}$ be the values between V_1 and V_n ($V_1 \leq V_2 \leq \cdots \leq V_{n-1} \leq V_n$). The new θ_1 is estimated as

$$\theta_1 = \frac{V_{i^*} + V_{i^*+1}}{2}$$

$$V_{i^*} = \underset{V_i}{\operatorname{argmin}} (\text{no. of } V_j + \text{no. of } V_k) \tag{3}$$

where V_k is a value of a labeled positive example, V_j is a value of a labeled negative example, and $V_1 \leq V_k \leq V_i$, $V_{i+1} \leq V_j \leq V_n$.

If $SP > LN$, θ_1 will completely discriminate the labeled positive from negative examples. Otherwise, θ_1 will give the minimum errors of misclassified examples.

2.1.2. *Naïve Bayes Classifier (Classifier2)*

For text classification, naïve Bayes is among the most commonly used and the most effective methods.[14] To represent text, the method usually uses *bag-of-words* representation. Instead of bag-of-words, we use the simpler *bag-of-characters* representation in the problem of classification of Thai/non-Thai pages. This representation is suitable for a Web robot to identify Thai Web pages because it requires no word segmentation and thus it is very fast. In spite of its simplicity, our results show the effectiveness of bag-of-characters representation in identifying Thai Web pages, as shown later in Section 4.

Given a set of class labels $L = \{l_1, l_2, \ldots, l_m\}$ and a document d of n characters $(c_1, c_2, \ldots, c_n)$, the most likely class label l^* estimated by naïve Bayes is the one that maximizes $\Pr(l_j|c_1, \ldots, c_n)$:

$$l^* = \underset{l_j}{\operatorname{argmax}} \Pr(l_j|c_i, \ldots, c_n)$$

$$= \underset{l_j}{\operatorname{argmax}} \frac{\Pr(l_j)\Pr(c_1, \ldots, c_n/l_j)}{\Pr(c_1, \ldots, c_n)} \tag{4}$$

$$= \underset{l_j}{\operatorname{argmax}} \Pr(l_j)\Pr(c_1, \ldots, c_n/l_j) \tag{5}$$

In our case, L is the set of positive and negative class labels. The term $\Pr(c_1, \ldots, c_n)$ in Equation 4 can be ignored, because we are interested in finding the most likely class label.

Because there are usually an extremely large number of possible values for $d = (c_1, c_2, \ldots, c_n)$, calculating the term $\Pr(c_1, c_2, \ldots, c_n|l_j)$ requires a huge number of examples to obtain reliable estimation. Therefore, to reduce the number of required examples and improve reliability of the estimation, assumptions of naïve Bayes are made.[14] These assumptions are (1) the conditional independent assumption, i.e., the presence of each character is conditionally independent of all other characters in the document given the class label and (2) an assumption that the position of a character is unimportant, e.g., encountering the character "a" at the beginning of a document is the same as encountering it at the end. Clearly, these assumptions are violated in real-world data, but empirically naïve Bayes has been applied successfully in various text classification problems.[8,12,19]

Using the foregoing assumptions, Equation 5 can be rewritten as

$$l^* = \underset{l_j}{\operatorname{argmax}} \Pr(l_j) \prod_{i=1}^{n} \Pr(c_i|l_j, c_1, \ldots, c_{i-1})$$

$$= \underset{l_j}{\operatorname{argmax}} \Pr(l_j) \prod_{i=1}^{n} \Pr(c_i|l_j) \tag{6}$$

This model also is called the *unigram* model because it is based on statistics about a single character in isolation.

The probabilities $\Pr(l_j)$ and $\Pr(c_i|l_j)$ are used as the parameter set θ_2 of our naïve Bayes and are estimated from the training data. The prior probability $\Pr(l_j)$ is estimated as the ratio between the number of examples belonging to the class l_j and the number of all examples. The conditional probability $\Pr(c_i|l_j)$ of seeing character c_i given class label l_j, is estimated by the following equation:

$$\Pr(c_i|l_j) = \frac{1 + N(c_i, l_j)}{T + N(l_j)} \tag{7}$$

where $N(c_i, l_j)$ is the number of times character c_i appears in the training set from class label l, $N(l_j)$ is the total number of characters in the training set for class label

l, and T is the total number of unique characters in the training set. Equation 7 uses Laplace smoothing (adding one to all the character counts for a class) to avoid assigning probability values of zero to characters that do not occur in the training data for a particular class.

2.2. Subclassifiers in ICT for the Classification of Course/Noncourse Home Pages

The problem of classification of Web pages into course/noncourse home pages is described in Ref. 2. The course home page gives information about the subjects taught at the university, course syllabus, instructor, class schedule, etc., and noncourse home pages are instructor home pages, department home pages, and student home pages. In this problem, each Web page contains two sets of features: (1) words appearing in the content of the page and (2) words appearing on the hyperlinks that link to that page. Therefore, each page can be viewed in two different ways, i.e., content-based features and hyperlink-based features. With these two feature sets, we construct two naïve Bayes classifiers: the first one (*Classifier1* in Table I) learns from hyperlink features and the second one (*Classifier2*) learns from content-based features. Both classifiers use the naïve Bayes algorithm, which is the same algorithm described in Section 3.1, except that for this problem it uses *bag-of-word* representation.

2.3. Subclassifiers in ICT for the Classification of University-Related Web Pages

The goal of this classification problem is to classify Web pages into four categories, which are course, faculty, project, and student Web pages. In this problem, each Web page contains two sets of features: (1) words appearing in the content of the page and (2) words appearing in headings indicated by the hypertext markup language (HTML) tags (e.g., <H1>, <H2>, or <H3>). Therefore, each page can be viewed in two different ways, i.e., content-based features and heading-based features. With these two feature sets, we construct two naïve Bayes classifiers; the first one learns from content-based features and the second one (Classifier 2) learns from heading-based features. Both classifiers use the naïve Bayes algorithm, which is the same algorithm described in the Section 3.1. This problem also uses *bag-of-word* representation.

3. OTHER CLASSIFIERS USED IN COMPARISON

In our experiment, we will compare ICT with the following classifiers:

(1) A supervised word segmentation classifier
(2) A supervised naïve Bayes classifier
(3) A co–training-style classifier

Table II. The co–training-style algorithm.

Given:
- A set LE of labeled training examples
- A set UE of unlabeled examples

Create a pool UE' of examples by choosing u examples at random from UE.
Loop until no examples left in UE:
  Use LE to estimate the parameter set θ_1 of *Classifier1*
  Use LE to estimate the parameter set θ_2 of *Classifier2*
  Allow *Classifier1* with θ_1 to label p-positive and n-negative examples from UE'
  Allow *Classifier2* with θ_2 to label p-positive and n-negative examples from UE'
  Add these self-labeled examples to LE
  Randomly choose $2p + 2n$ examples from UE to replenish UE'

Supervised word segmentation and supervised naïve Bayes classifiers used in our comparison are the same as those described in Section 3.1, except that they are trained by a large number of hand-labeled data. The Co–training-style classifier is described in the next section.

3.1. Co–Training-Style Classifier

The cotraining algorithm is described in Ref. 2. The idea of the algorithm is that an example can be considered in two different views. For example, a Web page can be partitioned into the words occurring on that page and the words occurring in hyperlinks that point to that page.[2] Either view of the example is assumed to be sufficient for learning. The algorithm consists of two subclassifiers, each of which learns its parameter sets from each view of the example.

Based on this idea, we construct a co–training-style algorithm for our problems. The algorithm is shown in Table II. The algorithm uses two subclassifiers: T2
Classifier1 and *Classifier2*. These two classifiers are the same as ones of ICT:

(1) In the case of classification of Thai/non-Thai pages, we view each Web page as a set of words occurring on that page and a set of characters occurring on the page. The word segmentation classifier (*Classifier1*) is used to learn from the view of the word representation, and the naïve Bayes classifier (*Classifier2*) is used for the character representation. The parameters θ_1 and θ_2 of *Classifier1* and *Classifier2* are estimated in the same way as described in Section 2.1.
(2) In the case of classification of course/noncourse home pages, we view each Web page as words occurring on that page and the words occurring in hyperlinks that point to that page. The page-based classifier *Classifier1* learns from words occurring on that page. The hyperlink-based classifier *Classifier2* learns from words occurring in the hyperlinks. For this problem, both *Classifier1* and *Classifier2* are naïve Bayes classifiers.
(3) In the case of classification of university-related Web pages, we view each Web page as words occurring on that page and words appearing in all headings of that page. The content-based classifier *Classifier1* learns from words occurring on that page. The heading-based classifier *Classifier2* learns from words occurring in all headings of the page.

Our co–training-style algorithm is slightly different from the original one in that our algorithm will consume all data in UE. This is done to provide a fair comparison with the other methods.

4. EXPERIMENTAL RESULTS

We conducted experiments to compare ICT with the other classifiers described in the previous section: supervised word segmentation classifier (S-Word), supervised naïve Bayes classifier (S-Bayes), and co–training-style classifier (CoTraining). This section describes the data set, the setting for each classifier, and the results of the comparison on three classification problems: (1) Thai/non-Thai page, (2) course/noncourse home page, and (3) university-related Web page classification.

4.1. The Results on the Thai/non-Thai Page Classification

In this section, we describe the data set and experimental setting for algorithms and the results as follows.

4.1.1. Data Set and Experimental Setting

We collected the data set by starting from four Web pages: a Japanese Web page,[a] two Thai Web pages,[b] and an English web page.[c] From each of these four pages, a Web robot was used to recursively follow the links within the page until it retrieved 450 pages.

Therefore, we have ~900 Thai pages because Thai pages may link to ones that are in English or other languages. We also have ~450 Japanese and 450 English pages. All of these pages were divided into three sets, denoted as A, B, and C, each of which contains 600 pages (about 300 Thai pages, 150 Japanese pages, and 150 English pages). Note that HTML markup tags were removed before the training and testing process. We used threefold cross-validation in all experiments for averaging the results.

The settings for the classifiers are as follows:

(1) For ICT, we ran the algorithm with both incremental and batch modes. We refer to incremental-mode ICT and batch-mode ICT as I-ICT and B-ICT, respectively. We used consistency checking for I-ICT and no consistency checking for B-ICT. No label data were given to B-ICT. The initial θ_{10} was set to 0.7. For I-ICT; we gave 18 hand-labeled pages as initial labeled data for the naïve Bayes classifier.

(2) For CoTraining, the values of the parameters of the classifier (in Table II) were set in a similar way as in Ref. 2. Because CoTraining requires a small set of correctly preclassified training data, we gave the algorithm with 18 hand-labeled pages. In our experiment, we set the values of $|UE|$, p, n, and u to 1182, 3, 3, and 115, respectively.

[a]http://www.yahoo.co.jp.
[b]http://www.sanook.com, http://www.pantip.com.

Table III. The precision (%), recall (%), and F_1 measure of the classifiers for the problem of Thai/non-Thai page classification.

Classifier	P (%)	R (%)	F_1
I-ICT (Word)	100.00	99.44	99.72
B-ICT (Word)	100.00	99.00	99.50
S-Bayes	100.00	99.00	99.50
B-ICT (Bayes)	100.00	98.89	99.44
I-ICT (Bayes)	99.55	99.33	99.44
CoTraining (Bayes)	100.00	98.89	99.44
S-Word	99.08	99.61	99.34
CoTraining (Word)	100.00	98.66	99.33

4.1.2. Results

To evaluate the performance of the classifiers, we use standard precision (P), recall (R), and F_1 measure[d] (F_1) defined as follows:

$$P = \frac{\text{no. of correctly predicted positive examples}}{\text{no. of predicted positive examples}}$$

$$R = \frac{\text{no. of correctly predicted positive examples}}{\text{no. of all positive examples}}$$

$$F_1 = \frac{2PR}{P + R}$$

The results are shown in Table III. In the table, "CoTraining(Bayes)" and "Co-Training(Word)" are the results of naïve Bayes and word segmentation classifiers of *CoTraining*, respectively. "B-ICT(Bayes)" and "B-ICT(Word)" are for naïve Bayes and word segmentation classifiers of ICT with the batch mode and "I-ICT(Bayes)" and "I-ICT(Word)" are those of the incremental mode. T3

As shown in the Table III, I-ICT(Word) gave the best performance according to F_1 measure, followed by B-ICT(Word), which gave a comparable performance to S-Bayes. The performance of B-ICT(Bayes) also was comparable with that of CoTraining(Bayes) and I-ICT(Bayes). Compared with the other classifiers, S-Word and CoTraining(Word) did not perform well.

Compared with supervised classifiers, the performance of ICT was comparable with that of S-Bayes and quite better than that of S-Word. The results indicate that our system can effectively use unlabeled examples and the two modules succeed in training each other. The reason that I-ICT(Word) gave better performance than B-ICT(Word) comes from the consistency-checking step during the classification processes. Although we did not include the details of running time of all classifiers, from the experiments we found that B-ICT ran much faster than I-ICT and Co-Training.

[d]The F_1 measure has been introduced by van Rijsbergen[17] to combine recall and precision with an equal weight.

4.2. The Results on the Course/Noncourse Home Page Classification

Now, we describe the data set and experimental setting and the results on the course/noncourse page classification problem.

4.2.1. Data Set and Experimental Setting

The data for this experiment is obtained via file transfer protocol (FTP) from Carnegie Mellon University.[c] It consists of 1051 Web pages collected from the computer science department Web sites at four universities: Cornell, University of Washington, University of Wisconsin, and University of Texas. These Web pages have been hand labeled into two categories. We consider the category "course home page" as the positive class and the other as the negative class. In this data set, 22% of the Web pages are course home pages. AQ: 6

Each example is filtered to remove words that give no significance in predicting the class of the document. Words to be eliminated are auxiliary verbs, prepositions, pronouns, possessive pronouns, phone numbers, digit sequences, dates, and special characters. We have 230 course Web pages and 821 noncourse Web pages. Each Web page has two views (page based and hyperlink based). The training set contains 172 course Web pages and 616 noncourse Web pages. Three positive examples and nine negative examples were selected randomly from the training data set to be the initial labeled data. Therefore, each data set contains 12 initial labeled examples, 776 unlabeled training examples, and 263 test examples. Then, we used threefold cross-validation for averaging the results.

The settings for the classifiers are as follows:

(1) For ICT, we ran the algorithm with both incremental and batch modes using consistency checking. As we have no domain knowledge to provide to the classifier for this problem, we gave three positive and nine negative examples as initial labeled data for ICT. The parameters p and n in Table I were set to 1 and 3, respectively.

(2) For CoTraining, the values of the parameters of the classifier (in Table II) were set in the same way as in Ref. 2. Because CoTraining requires a small set of preclassified training data, we gave the algorithm with three positive and nine negative examples. In our experiment, we set the values of $|UE|$, p, n, and u to 776, 1, 3, and 75, respectively.

4.2.2. Results

The experimental results are shown in Table IV. In Table IV, I-ICT(Content) and I-ICT(Hyperlink) stand for the page-based and hyperlink-based naïve Bayes classifiers of I-ICT, respectively, and B-ICT(Content) and B-ICT(Hyperlink) are those of B-ICT. CoTraining(Content) and CoTraining(Hyperlink) are page-based and hyperlink-based naïve Bayes classifiers of CoTraining algorithm, respectively. S-Bayes(Content) and S-Bayes(Hyperlink) are supervised naïve Bayes classifiers, T4

[c]The World Wide Knowledge Base (web-kb) project (http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-51/www/co-training/data/course-co-train-data.tar.gz). Carnegie Mellon University.

Table IV. The precision (%), recall (%), and F_1 measure of the classifiers for the problem of course/noncourse page classification.

Classifier	P (%)	R (%)	F_1
I-ICT(Content)	94.04	80.46	86.72
S-Bayes(Content)	77.48	94.35	85.09
S-Bayes(Hyperlink)	87.81	62.17	72.80
I-ICT(Hyperlink)	67.54	72.41	69.89
CoTraining(Hyperlink)	62.41	59.19	60.75
CoTraining(Content)	91.91	34.49	50.15
B-ICT(Content)	67.70	39.76	50.10
B-ICT(Hyperlink)	62.11	34.08	44.01

which classify Web pages based on words in Web pages and words in hyperlinks, respectively.

As shown in the Table IV, I-ICT(Content) gave the best performance followed by S-Bayes(Content), S-Bayes(hyperlink), I-ICT(Hyperlink), CoTraining(Hyperlink), and CoTraining(Content). The performance of B-ICT was lower than the others. Compared with the performance of B-ICT on Section 5.1, the results of B-ICT on this problem were not good. This is because of the fact that unlike B-ICT in Section 5.1, which was given knowledge in the form of the dictionary, B-ICT on this problem had no knowledge about the domain. In this problem, B-ICT received only a small set of labeled examples for building its initial parameter set. As shown by the results, this initial parameter set did not contain enough statistical information for labeling the whole examples in batch mode. However, when we ran the algorithm with incremental mode, with the help of consistency checking, I-ICT incrementally added a small set of examples on each round and gave improved results over B-ICT.

The reason that I-ICT(Content) gave better performance compared with S-Bayes is because I-ICT(Content) cooperated with I-ICT(Hyperlink) while S-Bayes used the single classifier. The performance of I-ICT(Hyperlink) was not as good as that of I-ICT(Content). This is because hyperlinks contain fewer words or sometimes it contains only a proper noun. Therefore, it is less capable of building an accurate classifier. The training technique of I-ICT also is an effective way because its performance was better than that of CoTraining, which uses a different training technique.

4.3. The Results on the University-Related Web Pages

In this section, we describe the data set and experimental setting for algorithms and the results as follows.

4.3.1. Data Set and Experimental Setting

This experiment used the WebKB data set,[5] which contains Web pages gathered from the department of computer science in four universities. These Web

pages have been hand labeled into four categories, which are course home page, faculty home page, project home page, and student home page.

In this data set, some categories are actually closely related, which make the classification more difficult. A course home page gives information about the subject such as the course outline, the class schedule, and reference books. A faculty home page is an instructor home page, which provides information about the instructor's research in the teaching course. A project home page actually is a research home page. A student home page is a personal home page of a student who studies at the university. AQ: 7

We have 220 course Web pages, 147 faculty Web pages, 81 project Web pages, and 533 student Web pages. Each example was filtered to remove words that give no significance in predicting to the class of the document as in Section 4.2. Then, the word stemming process was applied to each sample by using Porter the algorithm[16] in order to remove all suffixes and search for similar words based on the root word. Finally, we extracted all headings appearing in each Web page to be the feature of heading-based classifier. Therefore, each Web page can be viewed as the series of words appearing in the page's content and words appearing in all headings belong to the Web page.

For each category, 5% of all examples were used as initial labeled data. The training set was composed of 70% of all examples and 25% of all examples were used as the test set.

We assume that each Web page may belong to more than one category; therefore, the learning process was performed by class basis to get the knowledge for each class. For example, we train the classifier to learn the concept of the course Web page by considering the course Web page as positive examples, whereas examples from other classes are considered to be negative ones. We conducted experiments by using threefold cross-validation for averaging the results of each category. The parameters p and n of the CoTraining algorithm and ICT are set to 1 and 3, respectively.

4.3.2. Results

Tables V and VI show the results of all classifiers using the heading-based feature and the content-based feature, respectively. In the tables, S-Bayes stands for the supervised naïve Bayes classifier, I-ICT and B-ICT are the naïve Bayes classifiers of the ICT algorithm using incremental and batch modes, respectively. CoTraining is the result of the naïve Bayes classifier of the CoTraining classifier. T5-6

Considering the performance of classifiers in Table V, we found that I-ICT outperformed S-Bayes and CoTraining in all categories. The summary result in Table VI shows the explicit comparison of each classifier's performance. According to the F_1 measure, I-ICT(Heading) obtained the best performance with 73.83% followed by S-Bayes(Heading) and I-ICT(Content). Most classifiers using the heading feature obtained higher performance than classifiers using the content feature. It implies that the heading feature has more potential than the content feature in helping the classifier to build the correct model used in the classification process. Considering the feature set, we found that I-ICT obtained the highest

Table V. The precision (%), recall (%), and F_1 measure of classifiers using threefold cross-validation.

Category	Classifier	P (%)	R (%)	F_1
Course	I-ICT(Heading)	87.15	93.94	90.42
	S-Bayes(Heading)	88.51	84.24	86.32
	CoTraining(Heading)	87.91	82.42	85.08
	I-ICT(Content)	90.56	79.40	84.61
	S-Bayes(Content)	94.33	58.79	72.44
	CoTraining(Content)	82.54	49.69	62.03
	B-ICT(Content)	25.00	100.00	40.00
	B-ICT(Heading)	25.00	100.00	40.00
Faculty	I-ICT(Heading)	77.96	66.67	71.87
	CoTraining(Heading)	83.88	56.76	67.71
	S-Bayes(Heading)	77.43	43.24	55.49
	I-ICT(Content)	69.58	34.23	45.89
	B-ICT(Content)	25.00	100.00	40.00
	B-ICT(Heading)	25.00	100.00	40.00
	CoTraining(Content)	68.78	25.23	36.92
	S-Bayes(Content)	55.56	13.51	21.73
Project	I-ICT(Heading)	66.10	73.27	69.50
	S-Bayes(Heading)	87.41	45.15	59.54
	I-ICT(Content)	56.82	47.60	51.80
	B-ICT(Content)	25.00	100.00	40.00
	B-ICT(Heading)	25.00	100.00	40.00
	CoTraining(Content)	53.21	24.22	33.29
	S-Bayes(Content)	93.33	14.80	25.55
	CoTraining(Heading)	49.70	15.07	23.13
Student	I-ICT(Heading)	87.51	41.28	56.10
	B-ICT(Content)	25.00	100.00	40.00
	B-ICT(Heading)	25.00	100.00	40.00
	CoTraining(Heading)	91.67	21.46	34.78
	S-Bayes(Heading)	97.92	19.09	31.95
	I-ICT(Content)	100.00	15.39	26.67
	CoTraining(Content)	80.95	12.30	21.36
	S-Bayes(Content)	100.00	2.50	4.88

Table VI. The precision (%), recall (%), and F_1 measure on the average performance of classifiers using threefold cross-validation.

Classifier	P (%)	R (%)	F_1
I-ICT(Heading)	79.68	68.79	73.83
S-Bayes(Heading)	87.81	47.93	62.01
I-ICT(Content)	79.24	44.16	56.71
CoTraining(Heading)	78.29	43.93	56.28
CoTraining(Content)	71.37	27.86	40.07
B-ICT(Heading)	25.00	100.00	40.00
B-ICT(Content)	25.00	100.00	40.00
S-Bayes(Content)	85.81	22.40	35.53

tapraid5/8c-int/8c-int/8c1203/8c0419-03a | franklim | S=4 | 9/22/03 | 7:18 | Art: 3049L | Input-bsu(bsu)

performance in both features, I-ICT also outperformed S-Bayes because I-ICT combines two classifiers based on different feature sets. For this classification problem, batch-mode ICT did not perform well because it has no domain knowledge, whereas incremental mode of ICT using different labeling technique obtained higher performance than other classifiers.

5. DISCUSSION AND RELATED WORK

We have applied ICT on three classification problems. The problem of Thai/non-Thai page classification is simpler than the problems of course/noncourse home page classification and university-related Web page classification. This can be seen by the performance of all classifiers, which decrease on the second and third problems. For a difficult problem, incremental-mode ICT seems to be more suitable than batch-mode ICT. Batch-mode ICT has an advantage because it runs fast and it is suitable for the problem where we can provide domain knowledge.

Although the performance of our method is comparable or better than the other classifiers, the precision and recall on the problem of course/noncourse and university-related Web page classification are still not high. This may be because of the simple model of the classifiers, i.e., naïve Bayes classifiers. We plan to construct some domain knowledge for giving to the classifier and uses more powerful classifiers to test in this problem in the near future.

Our technique is related to the expectation-maximization algorithm.[6] The EM algorithm is an effective method for dealing with missing values in data and has been applied successfully to text classification.[15] Nigam et al.[15] have shown that the accuracy of classifiers can be improved by using EM to augment a small number of labeled data with a large set of unlabeled data.

Meta-bootstrapping is another unsupervised algorithm for learning from unlabeled data.[9] Like our method, the algorithm is composed of two sublearning algorithms. However, the training process of meta-bootstrapping and the way of using data are different from our method. This algorithm is a multilevel algorithm and is very useful, especially in the complex domain where sublearning algorithms alone could not produce enough good results. We also plan to study this kind of multilevel algorithm for using with our method.

6. CONCLUSION

We have presented a method that effectively uses unlabeled examples to estimate the parameters of the system for classifying Web pages. The method is based on two subclassifiers that iteratively train each other. Because ICT consists of two subclassifiers, the algorithm has an ability to use the different feature sets of the Web pages during the learning process to construct the most appropriate model used in classifying unseen Web pages. Moreover, the ICT algorithm can be applied to other classification problem as well. With no prelabeled or a small set of prelabeled examples, our method gives high precision and recall on classifying Web pages. The performance of our method is competitive with those of supervised ones, which establishes the successful use of unlabeled data of our method.

Acknowledgments

This work is supported by the Thailand Research Fund and National Electronics and Computer Technology Center.

References

1. Apte C, Damerau F. Automated learning of decision rules for text categorization. ACM TOIS 1994;12(2):233–251.
2. Blum A, Mitchell T. Combining labeled and unlabeled data with co-training. In: Proc of the 11th Annual Conf on Computational Learning Theory, 1998. AQ: 8
3. Cohen WW. Fast effective rule induction. In: Proc of 12th Int Conf on Machine Learning. San Mateo, CA: Morgan Kaufmann; 1995. AQ: 9
4. Cohen WW, Singer Y. Context-sensitive learning methods for text categorization. ACM Trans Infn Syst 1998;17(2):141–173.
5. Craven M, DiPasquo D, Freitag D, McCallum A, Mitchell T, Nigam K, Slattery S. Learning to extract symbolic knowledge from the World Wide Web. In: AAAI-98, 1998. AQ: 10
6. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc B 1977;39(1):1–38.
7. Joachims J. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In: Proc of the 14th Int Conf on Machine Learning. San Mateo, CA: Morgan Kaufmann; 1997. pp 143–151. AQ: 11
8. Joachims T. Text categorization with support vector machines: Learning with many relevant features. In: Proc of the 10th Eur Conf on Machine Learning. New York: Springer Verlag; 1998. AQ: 12
9. Jones R, McCallum A, Nigam K, Riloff E. Bootstrapping for text learning tasks. IJCAI-99 Workshop on Text Mining: Foundations, Techniques and Applications. 1999. pp 52–63. AQ: 13
10. Lewis D. Naive (Bayes) at forty: The independence assumption in information retrieval. In: Proc of the 10th Eur Conf on Machine Learning. 1998. AQ: 14
11. Liere R, Tadepalli P. Active learning with committees for text categorization. In: Proc of the 14th Natl Conf on Artificial Intelligence. 1997. pp 591–596. AQ: 13
12. McCallum A, Rosenfeld R, Mitchell T, Nigam A. Improving text classification by shrinkage in a hierarchy of classes. Proc of the 15th Int Conf on Machine Learning. San Mateo, CA: Morgan Kaufmann; 1998. pp 350–358. AQ: 15
13. Meknavin S, Charoenpornsawat P, Kijsirikul B. Feature-based Thai word segmentation. In: Proc of Natural Language Processing Pacific Rim Symposium '97, 1997. AQ: 14
14. Mitchell T. Machine learning. New York: McGraw-Hill; 1997. pp 180–184.
15. Nigam K, McCallum A, Thrun S, Mitchell T. Text classification from labeled and unlabeled documents using EM. Mach Learning 2000. In press. AQ: 16
16. Porter MF. An algorithm for suffix stripping. 1980. pp 130–137. AQ: 17
17. van Rijsbergen CJ. Information retrieval. London: Butterworths; 1979.
18. Yang Y. An evaluation of statistical approaches to text categorizing. Inf Retriev J 1999. AQ: 18
19. Yang Y, Pederson J. Feature selection in statistical learning of text categorization. In: Proc of the 14th Int Conf on Machine Learning. San Mateo, CA: Morgan Kaufmann; 1997. pp 412–420. AQ: 19

THE EFFECTS OF DIFFERENT FEATURE SETS ON THE WEB PAGE CATEGORIZATION PROBLEM USING THE ITERATIVE CROSS-TRAINING ALGORITHM

Nuanwan Soonthornphisaj and Boonserm Kijsirikul
Machine Intelligence & Knowledge Discovery Laboratory
Department of Computer Engineering
Chulalongkorn University,
Phathumwan, Bangkok, 10330, Thailand.
Email : nuanwan@mind.cp.eng.chula.ac.th , boonserm@mind.cp.eng.chula.ac.th

Keywords: Web page categorization, Iterative Cross-Training, Feature sets

Abstract: The paper presents the effects of different feature sets on the Web page categorization problem. These features are words appearing in the content of a Web page, words appearing on the hyperlinks, which link to the page and words appearing on every headings in the page. The experiments are conducted using a new algorithm called the Iterative Cross-Training algorithm (ICT) which was successfully applied to Thai Web page identification. The main concept of ICT is to iteratively train two sub-classifiers by using unlabeled examples in crossing manner. We compare ICT against supervised naïve Bayes classifier and Co-Training classifier. The experimental results show that ICT obtains the highest performance and the heading feature is considerably succeed in helping classifiers to build the correct model used in the Web page categorization task.

1. INTRODUCTION

Nowadays, there is a massive increase of Web pages in the Internet. An ideal search engine should have the most updated information of all Web pages to provide the best search result for the user. Therefore, it should have an effective Web robot which crawls the Web and automatically classifies Web pages into categories, since Web page classification task is a tedious job and time consuming process if it is done by human. Thus, we want it to be automatic with a reliable classification result.

The problem of text classification has been explored by many researchers with variety of learning algorithms (Cohen & Singer, 1999; Jochim, 1998) When we give a sufficient set of labeled training examples, supervised learning is the most effective method for the classification. However, the construction of hand-labeled data must be done by a human and thus this is a painfully time-consuming process.

Though it is costly to construct hand-labeled data, in some domains it is easy to obtain unlabeled ones, such as data in the Internet. Therefore, we propose a new learning algorithm called incremental iterative cross-training (incremental-ICT) in order to utilize the available unlabeled data.

Our incremental-ICT is based on the ICT algorithm that has been successfully applied for identifying Thai Web pages (Kijsirikul et al., 2000). ICT employs two sub-classifiers to iteratively train each other by using unlabeled examples in crossing manner. ICT is based on the assumption that one of the sub-classifiers has some knowledge about the domain. However, this assumption is violated on some domains where we cannot give domain knowledge to the classifier. In such a problem, ICT does not perform well. In this paper, we propose a new algorithm, called incremental-ICT, which requires no such assumption.

To evaluate the robustness of our algorithm, we apply it to a more difficult problem than Thai Web page identification. The problem we are interested in is the classification of Web pages into course or non-course pages. Since the concept of ICT needs two classifiers, we build each classifier based on different feature sets using naïve Bayes classifiers.

We run experiments to evaluate the effectiveness of our method and to see the contribution of each feature set. In the experiments, we compare our method with the *Co-Training* algorithm (Blum &

J. Filipe et al. (eds.), Enterprise Information Systems III, 132–138.

Mitchell, 1998) and a supervised learning algorithm which uses a naïve Bayes classifier as a classification mechanism. The results show that incremental-ICT gives better performance than the other classifiers.

The paper is organized as follows. Section 2 presents feature sets used in the experiments. Section 3 describes our learning algorithm, and gives the details of a naïve Bayes classifier. Section 4 and 5 describes other learning methods used in our comparison. Section 6 describes the experimental results. Finally, Section 7 concludes our work.

2. FEATURE SETS

For the classification problem, the classifier's performance usually depends on the classification mechanism with the support of feature sets. The appropriate feature sets will help the classifier to enhance its classification correctness. Therefore we try to investigate the possible feature sets to see their contribution on the precision and recall of the classifier. Feature sets that we study are as follows.

2.1 Hyperlink

The Web page in the Internet is a special document. It has a unique characteristic which makes it different from other plain text documents. Most Web pages have hyperlinks that act like a pointer pointing to other pages and also have links from other pages pointing to them. In our case, we use the hyperlink which link to the page to be the first feature set.

2.2 Content

The content of a Web page provides information to the user in detail. We extract all words in the content to be the second feature set.

2.3 Heading

The heading phrase normally represents the main idea of the following content. We use this opportunity to extract all headings in the page in the hope that they could represent the main concept of a Web page.

3. INCREMENTAL ITERATIVE CROSS-TRAINING

The architecture of our learning algorithm consists of two naïve Bayes classifiers, each of which learns from different features of a Web page. For the ease of explanation, we will use the concrete example of feature sets which are words on hyperlinks linking to the page (hyperlink-based) and words on the page (content-based). Starting with a small number of labeled data, each classifier estimates its parameters and uses the learned parameters to classify unlabeled data for the other as shown in Figure 1. The classification for unlabeled data is done in incremental way, i.e., the algorithm incrementally labels a small number of data. The training data is duplicated into two sets: *TrainingData1* for training the hyperlink-based classifier and *TrainingData2* for training the content-based one. The concept of our algorithm is that if we could obtain reliable statistical information from the first classifier, it should be useful in classifying training data for the second classifier. After receiving training from each other, the parameters of the classifiers should be more reliable every iteration.

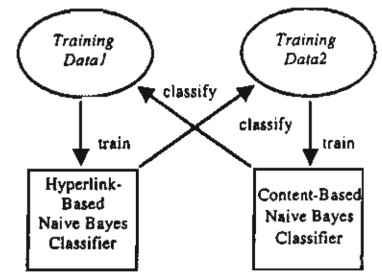


Figure 1: The architecture of iterative cross-training

The training algorithm of incremental-ICT is shown in Table 1. As shown in the table, the training process starts with the parameter estimation of both classifiers, i.e. hyperlink-based and content-based, using initial labeled data. For each round of iteration, the content-based classifier with the current parameter, θ_c , will classify training data into positive and negative examples. Then it will ask for the confirmation from the hyperlink-based classifier that considers another view of each example to make decision about which class the example should be. If both classifiers agree with the same classification

result, the most confident p positive and n negative examples will be labeled.

Table 1: The Incremental-ICT algorithm

Given:

- Two training sets *TrainingData1* of hyperlink-based data and *TrainingData2* of content-based data (*TrainingData1* and *TrainingData2* both contain U labeled examples).
- Use labeled data in *TrainingData1* to estimate the parameter set θ_h of the hyperlink-based classifier.
- Use labeled data in *TrainingData2* to estimate the parameter set θ_c of the content-based classifier.
- Loop until all data are labeled.
  - Use the content-based classifier with current θ_c to classify *TrainingData1* into positive and negative examples.
    - Check consistency of the classification with the hyperlink-based classifier. Label the class for the most confident p positive examples and most confident n negative examples.
  - Train the hyperlink-based classifier by the labeled examples in *TrainingData1* to estimate the parameter set θ_h of the classifier.
  - Use the hyperlink-based classifier with current θ_h to classify *TrainingData2* into positive and negative examples.
    - Check consistency of the classification with the content-based classifier. Label the class for the most confident p positive examples and most confident n negative examples.
  - Train the content-based classifier by the labeled examples in *TrainingData1* to estimate the parameter set θ_c of the classifier.

The hyperlink-based classifier is then trained by the labeled examples in *TrainingData1* to estimate the parameter set θ_h. With this current θ_h, the hyperlink-based classifier will classify *TrainingData2* into positive and negative examples. Then the consistency checking process is performed again to ask for the agreement from the content-based classifier. The most confident p positive and n negative examples will be labeled. The content-based classifier starts again with parameter estimation by using labeled examples in *TrainingData1*. These processes will be repeatedly done until all data are labeled.

The classification mechanisms of these two classifiers ar the same which use the naïve Bayes algorithm. This algorithm is a well-known approach and is considered to be one of the most effective way for text classification (Mitchell, 1997). The algorithm employs *bag-of-words* to represent the document. The method is described below.

Given a set of class labels $L = \{l_1, l_2, \ldots, l_m\}$ and a document d of n words $(w_1, w_2, \ldots, w_n)$, the most likely class label l^* estimated by naïve Bayes is the one that maximizes $Pr(l_j|w_1, \ldots w_n)$:

$$l^* = \underset{l_j}{argmax}\ Pr(l_j|w_1,\ldots,w_n) \tag{1}$$

$$= \underset{l_j}{argmax}\ \frac{Pr(l_j)Pr(w_1,\ldots,w_n|l_j)}{Pr(w_1,\ldots,w_n)} \tag{2}$$

$$= \underset{l_j}{argmax}\ Pr(l_j)Pr(w_1,\ldots,w_n|l_j) \tag{3}$$

For our data set, L is the set of positive and negative class labels which are course homepage and non-course homepage, respectively. $Pr(w_1,\ldots, w_n)$ in equation 2 can be ignored, as we are interested in finding the most likely class label. As there are usually an extremely large number of possible values for $d = (w_1, w_2, \ldots, w_n)$, calculating the term $Pr(w_1,\ldots,w_n|l_j)$ requires a huge number of examples to obtain reliable estimation. Therefore, to reduce the number of required examples and improve reliability of the estimation, assumptions of naïve Bayes are made. These assumptions are (1) the conditional independent assumption, i.e. the presence of each word is conditionally independent of all other words in the document given the class label, and (2) an assumption that the position of a word is unimportant, e.g. encountering the word "subject" at the beginning of a document is the same as encountering it at the end (Mitchell, 1997). Equation 3 can be rewritten as:

$$l^* = \underset{l_j}{argmax}\ Pr(l_j)\prod_{i=1}^{n} Pr(w_i|l_j,w_1,\ldots,w_{i-1}) \tag{4}$$

$$= \underset{l_j}{argmax}\ Pr(l_j)\prod_{i=1}^{n} Pr(w_i|l_j) \tag{5}$$

The probabilities $Pr(l_j)$ and $Pr(w_i|l_j)$ are used as the parameter sets θ_h and θ_c, and are estimated from the training data. The prior probability $Pr(l_j)$ is estimated as the ratio between the number of examples belonging to the class l_j and the number of all examples. The conditional probability $Pr(w_i|l_j)$, of seeing word w_i given class label l_j, is estimated by the following equation:

$$Pr(w_i|l_j) = \frac{1 + N(w_i,l_j)}{T + N(l_j)} \tag{6}$$

Where $N(w_i,l_j)$ is the number of times word w_i appears in the training examples from class label l_j. $N(l_j)$ is the total number of unique word in the training set. T is the number of class. Equation 6 employs Laplace smoothing (add one to all of word counts), to avoid assigning probability values of zero to words that do not occur in the training examples for a particular class.

To evaluate our method, we will compare it with the other two techniques that are the Co-Training and the supervised naïve Bayes classifiers. These classifiers are described in the following sections.

4. CO-TRAINING CLASSIFIER

The Co-Training algorithm explicitly uses the split of the features when learning from labeled and unlabeled data. Its approach is to build the naïve Bayes classifier for each of the distinct feature sets. Each classifier is initialized using a few labeled documents. Then every round of Co-Training, each classifier chooses the most confident p positive and n negative labeled examples to add to the labeled set of documents. The documents selected are those that have the highest posterior class probability, $Pr(l_j|d)$. Then, each classifier rebuilds from the augmented labeled set and the process repeats (Blum & Mitchell, 1998).

Table 2: The Co-Training algorithm

Given:

A set *LE* of labeled training examples

A set *UE* of unlabeled examples

Create a pool *UE'* of examples by choosing u examples at random from *UE*.

Loop while there exist documents without class labels:

- Use *LE* to estimate θ_h of the hyperlink-based classifier using the hyperlink portion of each document.
- Use *LE* to estimate θ_c of the content-based classifier using the page portion of each document.
- Allow the hyperlink-based classifier with current θ_h to label p positive and n negative examples from *UE'*.
- Allow the content-based classifier with current θ_c to label p positive and n negative examples from *UE'*.
- Add these self-labeled examples to *LE*.
- Randomly choose $2p+2n$ examples from *UE* to replenish *UE'*.

5. SUPERVISED NAÏVE BAYES CLASSIFIER

The basic concept of *supervised learning* for building a classifier is that it requires a set of examples with predefined classes. The classifier is then try to find some common properties of the different classes in order to make correct classification for unseen data. Thus, this kind of classifiers need a large number of labeled examples to correctly model the characteristic of the class during learning process. Labeling must be done by human to train the classifier accurately. In our experiment, we employ the naïve Bayes classifier as a supervised learning algorithm. The algorithm of the naïve Bayes is the same as one described in Section 3, except that it is trained by hand-labeled data.

6. EXPERIMENTAL RESULTS

In order to test the robustness of the incremental-ICT algorithm and to investigate the effectiveness of feature sets , we set up experiments on the problem of course/non-course Web page

classification, and compare the performance of incremental-ICT to the other classifiers, i e , the Co-Training algorithm and the supervised naïve Bayes classifier.

6.1 Data Set

The data for our experiments is obtained via ftp from Carnegie Mellon University (The World Wide Knowledge Base Project). It consists of 1,051 Web pages collected from Computer Science department Web sites at four universities: Cornell, University of Washington, University of Wisconsin, and University of Texas. These Web pages have been hand-labeled into two categories. We consider the category "course home page" as the positive class and the other as the negative class. In this dataset, 22% of the Web pages were course homepages and the rest were non-course homepages.

In this data set, the two Web categories are actually closely related which make the classification more difficult. A course home page gives information about the subject such as the course outline, the class schedule, reference books. A non-course homepage is an instructor homepage or department Web page.

6.2 Experimental Setting

We have 230 course Web pages and 821 non-course Web pages. Each sample is filtered to remove words which give no significance in predicting to the class of the document. Words to be eliminated are auxiliary verbs, prepositions, pronouns, possessive pronouns, phone numbers, digit sequences, dates and special characters. The training set contains 172 course Web pages and 616 non-course Web pages.

Three positive examples and nine negative examples were randomly selected from the training dataset to be initial labeled data. Therefore, each data set contains 12 initial labeled examples, 776 training examples and 263 testing examples. We then used 3-fold cross-validation (Mitchell, 1997) for averaging the results. Three positive and nine negative samples are used as the initial labeled data for the incremental-ICT and the Co-Training algorithms. The parameters p and n in Table 1 and Table 2 is set to 1 and 3, respectively.

6.3 The Results

Standard precision (P), recall (R), accuracy (A) and F1-measure (F1) are used to evaluate the performance of the classifiers. These are defined as follows.

$$P = \frac{\text{no. of correctly predicted positive examples}}{\text{no. of predicted positive examples}}$$

$$R = \frac{\text{no. of correctly predicted positive examples}}{\text{no. of all positive examples}}$$

$$A = \frac{\text{no. of correctly predicted examples}}{\text{no. of all examples}}$$

$$F1 = \frac{2PR}{P+R}$$

6.4 Experiment using content and hyperlink features

For the first experiment, we use words appearing in the content of a Web page as the feature for the first classifier. The second classifier uses words appearing on the hyperlink as a feature set. The results are shown in Table 3.

Table 3: Performance of content-based and hyperlink-based classifiers using 3-fold cross-validation: P = Precision, R = Recall, A = Accuracy, F1 = F1-measure.

Classifier	P(%)	R(%)	A(%)	F1
I-ICT (content)	94.04	80.46	94.55	86.72
S-Bayes (hyperlink)	85.34	63.22	89.48	72.61
I-ICT (hyperlink)	67.54	72.41	85.17	69.89
Co-Training (content)	81.52	54.08	87.32	65.03
S-Bayes (content)	99.05	42.20	87.25	58.97
Co-Training (hyperlink)	75.92	44.83	84.28	56.37

In Table 3, I-ICT (content) and I-ICT (hyperlink) stand for the content-based and hyperlink-based naïve Bayes classifiers of the incremental-ICT, respectively. Co-Training (content) and Co-Training (hyperlink) are content-based and hyperlink-based naïve Bayes classifiers of the Co-Training algorithm, respectively. S-Bayes (content) and S-Bayes (hyperlink) are supervised naïve-Bayes classifiers, which classify Web pages based on words in Web pages and words in hyperlinks, respectively.

As shown in the table, I-ICT (content) gives the best performance followed by S-Bayes (hyperlink), I-ICT (hyperlink), Co-Training (content), S-Bayes(content) and Co-Training (content), respectively. The reason that I-ICT (content) gives better performance compared to S-Bayes is because I-ICT (content) cooperates with I-ICT (hyperlink) while S-Bays uses single classifier. The performance of I-ICT (hyperlink) is not as good as that of I-ICT (content). This is because hyperlinks contain fewer words and thus are less capable of building the accurate classifier. The training technique of I-ICT is also an effective way, as its performance is better than that of Co-Training, which uses a different training technique.

6.5 Experiment using content and heading features

In order to see the impact of the heading feature on the categorization problem, we did experiments using heading-based classifier with various learning algorithms.

As shown in Table 4, I-ICT (content) and I-ICT(heading) stand for the content-based and heading-based naïve Bayes classifiers of the incremental ICT algorithm. S-Bayes (heading) and S-Bayes (content) are supervised naïve Bayes classifiers based on heading and content features, respectively.

Co-Training(heading) and Co-Training (content) are the heading-based and content-based naïve Bayes classifiers of the Co-Training algorithm.

Table 4: Performance of heading-based and content-based classifiers using 3-fold cross-validation: P = Precision, R = Recall, A = Accuracy, F1 = F1-measure.

Classifier	P(%)	R(%)	A(%)	F1
I-ICT (content)	98.12	78.66	95.05	87.32
S-Bayes (heading)	96.51	77.58	94.43	86.02
I-ICT (heading)	80.72	89.65	92.65	84.95
Co-Training (heading)	79.71	74.71	90.24	77.13
Co-Training (content)	82.49	43.68	85.93	57.11
S-Bayes(content)	99.05	42.20	87.25	58.97

The best performance belongs to the content-based naive Bayes classifier of I-ICT followed by the supervised naïve Bayes classifier based on the heading feature, the heading-based of I-ICT, the heading-based of co-training, the content-based of co-training and the content-based of the supervised naïve Bayes classifier.

7. DISCUSSION AND CONCLUSION

In this paper, we have demonstrated the concept of the I-ICT algorithm and investigate the impacts of feature sets to the classification correctness. From the experimental results, the performance of S-Bayes (heading) is much higher than that of S-Bayes (hyperlink). It means that the heading feature has more potential than the hyperlink feature in helping the classifier to build the correct model used in categorization task. This is because the detail of the Web page is usually organized into sub-sections with the headings, which represent the main idea of the following content. Thus the structure of all headings in a page should give some common properties that is useful to identify its category. In the contrary, the words in the hyperlink, which links to the page could not provide enough information to identify the class of the page. This is not surprising because normally the hyperlink phase contains just few words referring to the page. For the worst case, the hyperlink might contain only a proper noun that is not sufficient in classifying that referring page.

According to the F1-measure, we obtained only 58.97% accuracy for S-Bayes using the content feature. It implies that the content feature alone could not help much in Web page classification because the two Web categories actually have high relevance in detail. Therefore the naïve Bayes classifier could not find the exact model for each category using all words appearing in the page.

Considering all classification mechanisms, we found that our I-ICT algorithm provides the highest correctness in both experiments. This is because I-ICT combines two classifiers based on different feature sets and these two classifiers cooperate with each other during the training process. The I-ICT algorithm has been proved to be robust under new assumption that each example can be viewed in two different views using different feature sets. With the consistency checking process, which is used to compensate the lack of domain's knowledge of the classifiers, our algorithm outperformed the supervised naïve Bayes algorithm, and Co-Training algorithm.

ACKNOWLEDGEMENT

This paper is supported by the Thailand Research Fund and National Electronics and Computer Technology Center.

REFERENCES

Kijsirikul, B., Sasipongpairoege, P., Soonthornphisaj, N. and Meknavin, S., 2000, 'Supervised and Unsupervised Learning Algorithms for Thai Web Pages Identification', *Proceeding of the Pacific Rim*

International Conference on Artificial Intelligence (PRICAI-2000), 690-700

Blum, A. and Mitchell, T., 1998, 'Combining Labeled and Unlabeled Data with Co-Training', *Proceeding of the Eleventh Annual Conference on Computational Learning Theory*.

Cohen, W. and Singer, Y., 1999, 'Context-sensitive learning methods for text categorization', *ACM Transactions on Information Systems*, 17(2): 141-173.

Joachims, T., 1998, 'Text categorization with support vector machines: Learning with many relevant feature', *Proceedings Tenth European Conference on Machine Learning*, Springer Verlag

Nigam, K., McCallum, A., Thrun, S and Mitchell, T., 2000, 'Text classification from labeled and unlabeled documents using EM', *Machine Learning*, 39(2/3): 103-134.

Apte, C., and Damerau, F., 1994 'Automated learning of decision rules for text categorization', *ACM TOES*, 12(2): 233-251.

Mitchell, T., 1997, *Machine Learning*, McGraw-Hill, New York, 180-184.

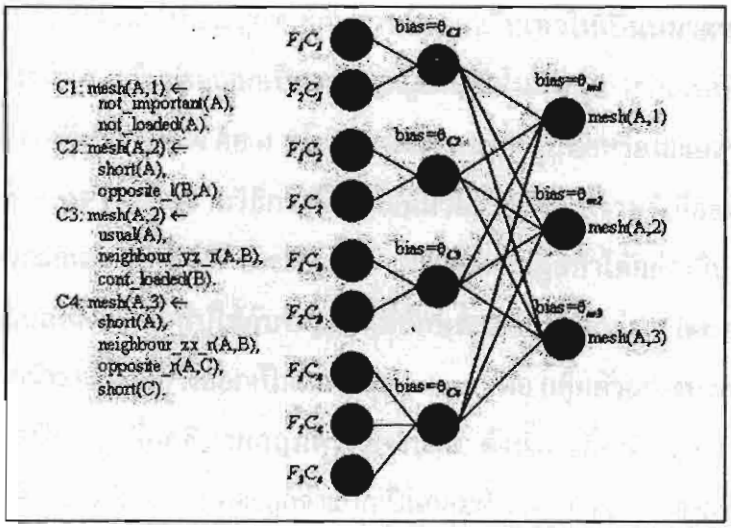
The World Wide Knowledge Base (web-kb) project, http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-51/www/co-training/data/course-cotrain-data.tar.gz, Carnegie Mellon University, U.S.A.

เอกสารหมายเลข 5

รางวัลผลงานวิจัย
จุฬาลงกรณ์มหาวิทยาลัย
ประจำปี ๒๕๔๕

รางวัลผลงานวิจัย
กองทุนรัชดาภิเษกสมโภช จุฬาลงกรณ์มหาวิ
ประจำปี ๒๕๔๕

ISBN 974-13-2291-7



ผลงานวิจัยเรื่อง การทำเหมืองเว็บไทยโดยเทคนิคการเรียนรู้ของเครื่องและการโปรแกรมตรรกะเชิงอุปนัย

Thai Web Mining Using Machine Learning and Inductive Logic Programming

โดย ผู้ช่วยศาสตราจารย์ ดร.บุญเสริม กิจศิริกุล

แหล่งทุนที่ได้รับ ทุนพัฒนานักวิจัย จากสำนักงานกองทุนสนับสนุนการวิจัย

5.3