Automatic Corpora Construction for Text Classification

Dandan Wang, Qingcai Chen, Xiaolong Wang, Bingyang Yu
Key Laboratory of Network Oriented Intelligent Computation
Harbin Institute of Technology Shenzhen Graduate School, China
{wangdandanhit, qingcai.chen}@gmail.com
wangxl@insun.hit.edu.cn
bingyang.yu@hotmail.com

Abstract

Since the machines become more and more intelligent, it is reasonable to expect the automatic construction of text classifiers by given just the objective categories. As trade-off solutions, existing researches usually provide additional information to the category terms to enhance the performance of a classifier. Unique from them, in this paper, we construct the standard corpora from the web by just providing text categories. Since there are millions of manually constructed websites, it is hopeful to find out proper text categorization (TC) knowledge. So we directly go to the web and use the hierarchies implied in navigation bars to extract and verify TC resources. By addressing the issues of navigation bar recognition and text filtering, the corpora are constructed for given text categories and the classifiers are trained based on them. We conduct our experiments on the large scale of webpages collected from the 500 top English websites on Alexa. The Open Directory Project (ODP) is used as testing corpus. Experimental results show that, being compared with the classifier based on manually labeled corpus, the classifier trained on auto constructed corpora reaches comparable performance for the categories that are well covered by the training corpus.

1 Introduction

As one of the key techniques in web information processing, text classification has been studied for a long time (Aas and Eikvil, 1999; Wang and Li, 2011; Yang and Pedersen, 1997). A growing number of machine learning techniques have been applied to text classification and some of them have proven to be successful (Miao and Kamel, 2011; Sebastiani, 2002). In the machine learning approach, the learning process is an instance of supervised or semi-supervised learning because classifier can be built automatically by learning from adequately pre-labeled training documents and then classified unseen documents (Feldman and Sanger, 2007). However, the task of manually labeling a large amount of documents is time-consuming and even impractical. Given a general classification task, people usually construct training data in two ways. One is augmenting a small number of labeled documents with large amounts of unlabeled ones to guide the learning model iteratively, so that the new classifier can label the unlabeled documents (Jiang, 2009; Nigam et al., 2000). In such studies, the bootstrapping technique is often used to label the unlabeled documents and refine the initial classifier (Gliozzo et al., 2009; Ko and Seo, 2009). The other is collecting training corpora from the Web. Such works use the class name and its associated terms to collect training corpora iteratively (Huang et al., 2005). Cheng (2009) and Day et al. (2009) firstly sample the Web with a set of given class names, and then query the keywords manually populated from each class by search engines for retrieving quality training documents. Huang et al. (2004) proposed a LiveClassifier system which also makes use of search engines for automatically constructing training classifier.

Though reached encouraging performance, above methods have some limitations in organizing training corpora. For the first method, although some algorithms just use a small set of labeled documents, which still require much time and effort for complicated categories (Chen et al., 2009); And the second method depends on several external resources, which greatly limits its flexibility and reliability. For example, manually given keywords or terms for a class are easily affected by

International Joint Conference on Natural Language Processing, pages 726–732,
Nagoya, Japan, 14-18 October 2013.

different persons; different search engines may also bring different results with various type of noises contained in search results (Huang et al., 2004).

Inspired by these issues, we design a new system to automatically acquire training corpora. Given a class hierarchy, our basic idea is to collect corpora merely based on class names. Firstly, we crawl the webpages starting with several selected websites and identify the navigation bars of these websites. Then each navigational item in the navigational bars is matched with the class names. The valid subpages from a navigational item are labeled with the matched class name. After extracting contents from these subpages, the initial candidate corpora are constructed. Finally clustering algorithm is used to remove noises from the corpora. In latter parts of paper, we denote the automatic constructed corpora as ACC.

The main contributions of this paper are:

(1) An automatic system for constructing classification corpora is built. It is a new way to collect large-scale, high quality corpora; moreover, it is completely adaptive to any kind of class hierarchy;

(2) To improve the ACC quality, text clustering based automatic noise filtering approaches are proposed and analyzed;

(3) The proposed system and methods are evaluated on large scale standard corpora and encouraging results are reached.

The remainder of this paper is organized as follows. Section 2 described the architecture of the automatic corpora construction system; Section 3 and 4 present experimental settings and results respectively. The paper is closed with conclusion and future work in section 5.

2 Automatic Corpora Construction

In this paper, we propose a novel system that can automatically acquire effective training data through web mining. The architecture of the system is given in Fig 1. It is composed of four modules: data collection, navigational processing, candidate corpora construction and corpora denoising. The data collection module crawls webpages from the given URL seeds. The navigational processing module is to extract the navigational bars from downloaded web pages, and to make category judgments for each navigational item. The candidate corpora construction module is to get the candidate corpora by performing content extraction for the valid links from the navigational items. The corpora denoising module makes clustering for all texts in the candidate corpora and removes the noisy web pages for each category.

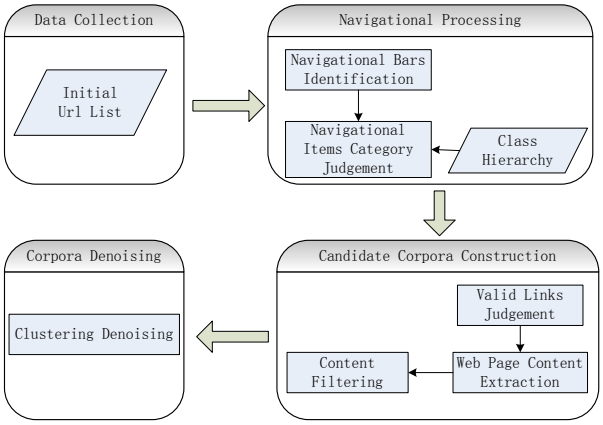

Figure 1: Architecture of the ACC System.

2.1 Navigational Bars Processing

Generally, the navigational bars locate in one block or several blocks of a web page. Thus it is necessary to split the web page into blocks for identifying the navigational bars. In this paper, we use the approach proposed by (Keller and Nussbaumer, 2012) for segmentation and simplify it by three rules as follows.

(1) Remove leaf nodes which are not link nodes;

(2) If the node is the only child of of its parent node, delete the parent node and directly connect the node with its ancestor node;

(3) If the node has two children nodes, and the first child node is the link node, while the other is not, then delete the node and connect the two children nodes with its ancestor node.

In latter parts of this paper, the set of blocks after segmentation is denoted as PageSec.

2.1.1 Navigational Bars Identification

The approaches for navigational bars identification can be divided into two categories, i.e., rule-based and graph-based filtering respectively. In this paper, we compared two different approaches for the navigational bars identification.

Rule-based Identification: Firstly, we conduct filtering for each block in PageSec. The filtering conditions include link type, link uniqueness etc. Then a ranking formula is constructed to calculate the score for each blocks. The features used in the formula are listed as follows.

(1) Consistent degree of link depth between the anchor texts within block: Each anchor text

within the block will point to a link. The depth of a link is represented by its slashes' number. The consistent degree of link depth between the anchor text within block is calculated by formula (1).

$$Dep\,(Sec) = \frac{-\sum_{i=1}^{n}\left(P\left(AcD_i\right)\ln\left(P\left(AcD_i\right)\right)\right)}{\ln\left(n\right)} \quad (1)$$

Where n: The number of anchors with different depth within the block.
$P\left(AcD_i\right)$: The proportion of anchors with the ith anchor depth within the block.

(2) Consistent degree of word number between the anchor texts within block: Generally, the word number of each navigational item is consistent. The more tidy appearance of the items, the more likely to belong to the same navigational bar. The consistent degree of word number between the anchor texts within block is calculated in the same way with the consistent degree of link depth.

(3) The proportion of the remaining anchor texts within block.

Integrating the three features above, we construct a linear weighted summation formula to rank for the blocks. Finally, the top-ranking blocks with score bigger than a threshold are added to the candidate of the navigational bars.

Graph-based Identification: Firstly, we construct the relationship graph of links and find the maximum complete subgraph by the Bron-Kerbosch algorithm. Then we use a discriminant algorithm to extract the final navigational bars.

(1) Construction of Links Relationship Graph: Each web page is represented with a node, if page A has a link pointing to page B, a directed edge from A to B is generated. We deleted single directional edges and preserved bi-directional edges; moreover, the directed graph is transformed to be undirected for simplification. In this paper, the Bron-Kerbosch algorithm is used to get the maximum complete subgraph.

(2) Extraction of Navigational Bars: For extracting navigational bars, we need to take advantage of maximum complete subgraphs to conduct filtering on the block structure of the web page. The pseudo code is listed in algorithm1. In addition, as the vertex number increases to a value, the running time of searching for the maximum subgraphs becomes unacceptable. Thus we apply an approximate algorithm to extract the navigational bars when the vertex number is more than 100.

2.1.2 Navigational Items Category Judgment

After identifying the navigational bars, we need to match the navigational items with each class of the given class hierarchy. The cosine similarity in the vector space model is used for calculating the matching degree. Moreover, in order to get a better result, we apply stemming for the navigational items and word expansion for the class names. Given a navigational item, we firstly calculate its similarities with all classes and sort those classes in descending order of the similarities. If the maximum similarity in the rank is unique and greater than a given threshold, label the navigational item with the corresponding class. Otherwise, we use the URL information to make further judgment.

Algorithm1: Navigational bars extraction
Input: Maximum complete subgraph MCSQueue
Block set of homepage PageSec
Output: Candidate navigational bars CandNav
Step1: Label all the elements in MCSQueue with unprocessed.
Step2: Select an unprocessed subgraph SGraph from MCSQueue. If all processed, turn to step 4.
Step3: Filter on all the elements in PageSec, remove the elements not in SGraph and save the result in CandNav, turn to step 2.
Step4: Sort all blocks in CandNav from more to less by the number of elements.
Step5: Traverse CandNav from the beginning and delete Sec if current block contains all the elements of block Sec which is at the position behind.
Step6: End

2.2 Candidate Corpora Construction

2.2.1 Valid Links Judgment

A navigational item within the navigational bar usually points to a hub web page that is mainly composed of a set of links, which makes it difficult to extract relevant web pages for a given topic. In general, the links that point to external websites are treated as invalid links and can be directly filtered. Since some kind of invalid links, such as the *Login, Sitemap* etc, occur many times in the website and the times of category assignment to the invalid links are significantly more than the valid links. In this paper, we regard the links with the times of category assignment more than a threshold as invalid links. Since there are multiple nav-

igational paths that can lead to a web page, each valid link may have multiple category labels and the voting strategy is used to label the valid link.

2.2.2 Web Page Content Extraction

Extracting content from webpages has been researched for decades and numerous methods have been proposed. Tim proposes a method to extract content text from diverse web pages by using the HTML documents tag ratios (Weninger et al., 2010). Sun presents Content Extraction via Text Density (CETD) to extract content and also proposes a method called DensitySum to extract integral content. Moreover, CETD-DS has shown that it is an effective and robust content extraction algorithm (Sun et al., 2011). In this paper, we apply Tim's tag ratio to extract content.

Tag Ratio: Tim's tag ratio is the ratio of HTML tags characters and Non-HTML tags characters at each row of the HTML source code. It can be described by formula (2) as follows:

$$TagRatio_i = \frac{NonTagChars_i}{TagNum_i} \quad (2)$$

Where $NonTagChars_i$: The number of Non-HTML tags characters at the ith row.
$TagNum_i$: The number of HTML tags at the ith row. For the hyperlink tag, multiply it by 2.

Since the comments, scripts and CSS tags do not contain the text content, we remove them from the HTML code while calculating the tag ratio. For the tag ratio, we apply the standard gaussian approach to make smoothing. Firstly we construct a gaussian kernel (Keerthi and Lin, 2003) with radius of 1 and variance of 2 by formula (3).

$$k_i = \sum_{j=-\lceil\sigma\rceil}^{\lceil\sigma\rceil} e^{\frac{-j^2}{2\sigma^2}}, 0 \le i \le 2\lceil\sigma\rceil \quad (3)$$

After normalization, we get (4):

$$k'_i = \frac{k_i}{\sum_{j=0}^{2\lceil\sigma\rceil} k_j}, 0 \le i \le 2\lceil\sigma\rceil \quad (4)$$

Then we make the convolution operation with this filter and tag ratio to get the smoothing tag ratio.

Content Extraction: Most of the methods for content extraction try to use different threshold selection strategies for different features, but the generalization performance of these algorithms are still not satisfactory. Therefore, we adopt the K-means clustering with two-dimensional features to get the final content. The first dimensional feature is the smoothing tag ratio, the second one is the approximate derivative of the tag ratio.

2.3 Corpora Denoising

Ideally, web pages with the same category label in the candidate corpora belong to the same topic. However, because of the different website authority and management level, some web pages which do not belong to the category topic are also classified into this category. In addition, some irrelevant web pages are preserved owing to the weakness of the valid links judgment. These noisy web pages greatly reduce the quality of the candidate corpora. Therefore, we apply K-means clustering algorithms to conduct corpora denoising. After clustering, the large clusters are preserved while the smaller ones are removed as the noises.

3 Experimental Settings

3.1 Databases

Self-collected Data: A universal crawler is implemented for the automatic corpora construction task. In addition to the general crawler task, it records the jumping information between web pages for locating the target web pages and the orders of visiting each page in the crawling process. The initial seed links of the crawler are collected from the Alexa[1] top 500 websites of each category on May 26, 2012. After removing the duplicated websites, there are 5593 ones left. Take them as the starting links and perform crawling, finally we crawled a total of 783035 web pages.

ODP (Open Directory Project): ODP is one of the largest corpora for web page classification. We download the ODP corpus on April 22, 2012. There are totally 4066266 web page links in which 2144930 web page links are downloaded.

Dataset Split: In our experiments, both the automatic constructed data and ODP are randomly split into three near equal scale of subsets with two parts for training and the remaining for testing.

3.2 Experimental Settings

Preprocessing: The class hierarchy of ODP is used for the automatic corpora construction. Libxml2 is used to parse the web pages into the DOM trees. For the web pages that cannot be directly parsed, tidy is applied to correct the syntax mistakes. The Porter algorithm (Jongejan and

[1] Alexa:http://www.alexa.com/

Dalianis, 2009) is used for stemming. Generally, there are two kind of ways to make word expansion. One is based on WordNet (Fellbaum, 1998) and the other is to obtain the description words for categories by search engine or other external resources. In this paper, the description words for each category come from its secondary classification keywords under the ODP class hierachy.

Classification and Clustering: Expected Cross Entropy is selected for feature selection. 8000 dimensional features are selected out. LibSVM [2] is used for classification where the linear kernel and default settings are applied. K-means is used for clustering where K is set to 8.

4 Experimental Results

4.1 Performance of ACC system

Navigational Bars Identification: To evaluate the performance of navigational bars identification, we randomly select five websites from the initial seed links for human label. The results of rule-based and graph-based navigational bars identification are shown in Table 1. N_{nb} denotes the number of navigational bars. A_{nr} and R_{nr} denote the accuracy and recall of rule-based identification approach. A_{ng} and R_{ng} denote the accuracy and recall of graph-based identification approach. We can see that both methods reach high accuracy and relatively lower recall without significant difference. The reason is that some external links are filtered while some navigational bars are composed of items from different domains.

Navigational Items Category Judgment: We randomly sampled 100 websites for human label from the initial seeds of the crawler. The performance of navigational items category judgment is demonstrated in Table 2. Since the accuracy of category judgment is very high, a navigational identification method with higher recall is selected to increase the size of the final corpora.

Web Page Content Extraction: We evaluate the performance of web page content extraction on seven standard corpora: CleanEval-Eng, NY Times, Yahoo!, Wikipedia, BBC, Arts Technica and Chaos. Table 3 shows that the web page content extraction algorithm performs well on accuracy for all the seven corpora, but the recall is not satisfactory on corpora like Wikipedia and Yahoo!

[2]Liblinear:http://www.csie.ntu.edu.tw/ cjlin/liblinear/

Website	N_{nb}	$A_{nr}(\%)$	$A_{ng}(\%)$	$R_{nr}(\%)$	$R_{ng}(\%)$
CNN	49	65	81.3	53.1	53.1
Yahoo!	41	36.2	52.9	51.2	22.0
EatingWell	81	87.2	78.5	43.6	79.5
Adobe	39	97.5	65.5	36.4	17.8
Cornell	38	100	51.6	65.8	42.1

Table 1: Performance of navigational bars identification

Category	**Judgement Accuracy**
Arts	31%
Business	100%
Computers	93%
Games	85%
Health	91%
News	88%
Science	100%
Shopping	83%
Society	87%
Sports	98%

Table 2: Category judgment performance of navigational items

Corpora	**Accuracy**	**Recall**	**F1**
CleanEval-Eng	85.91%	75.42%	80.32%
NY Times	84.24%	84.90%	84.57%
Yahoo!	95.40%	66.29%	78.22%
Wikipedia	98.82%	34.82%	51.50%
BBC	93.74%	83.55%	88.35%
Arts Technica	96.23%	92.16%	94.15%
Chaos	86.08%	94.47%	90.08%

Table 3: Performance of web page content extraction on 7 corpora

Corpora Accuracy: In Table 4, we demonstrate the performance of ACC on each category. N_p denotes the number of crawled web pages. A denotes the accuracy before clustering and A_c denotes the accuracy after clustering. This table shows that the macro-average accuracy can reach 68.18%. Furthermore, the applying of simple text clustering method contributes up to 2.73% accuracy performance gains, which demonstrates the effectiveness of our denoising method.

4.2 Performance of Classification

Size of ACC and ODP: The number of crawled web pages and left web pages after content extrac-

tion in ACC and ODP denoted by N_p and N_{cp} respectively are given in Table 5. From this table we can see that only a little part of crawled web pages in both data sets contain effective contents. It is because that most of the web pages in ODP are the homepages of websites, and many web pages mainly contain pictures rather than effective contents in ACC.

Classification Results of ACC and ODP: In Table 6, we show the overall SVM classification performance of each category on ODP and ACC. The macro-F1 is approximately 72.3% and 79.8% respectively. By comparing the classification results of ODP and ACC on each category, we can see that the performance of ACC is better than ODP on accuracy and recall.

Category	N_p	A	A_c
Arts	**32773**	**41%**	**40%**
Business	10477	86%	85%
Computers	2752	60%	65%
Games	20280	73%	76%
Health	7875	52%	59%
News	33850	70%	72%
Rigional	2256	70%	74%
Science	2358	71%	70%
Shopping	7820	57%	64%
Society	7381	77%	81%
Sports	14769	63%	64%
Average	—	**65.45%**	**68.18%**

Table 4: Performance of ACC on each category

Category	N_p		N_{cp}	
	ACC	ODP	ACC	ODP
Arts	32773	206887	1327	5043
Business	10477	208069	1167	4641
Computers	2752	95136	55	1487
Games	20280	44800	357	876
Health	7875	52230	252	977
News	33850	7438	1179	376
Science	2358	91841	136	2735
Shopping	7820	71070	103	853
Society	7381	161806	278	4304
Sports	14769	74317	742	1985

Table 5: Size of ACC and ODP for each category

But it is clear that this can not prove ACC is superior to ODP. Finally we train a classifier with all the automatic constructed corpora and use 1/3 proportion of ODP for testing. The results are shown in Table 7. The results show that the classification performance is relatively poor. The main reason is that the coverage of ACC is inadequate, which leads to a big distribution difference between the training and testing data. The factors that influence the coverage of corpora include the size of initial URL seeds, the crawling depth, the recall of navigational bars etc.

Class	Accuracy(%)		Recall(%)		F1(%)	
	ODP	ACC	ODP	ACC	ODP	ACC
Arts	74.1	71.6	84.2	77.3	78.8	74.3
Busi	60.9	76.7	65.4	81.9	63.1	79.2
Comp	73.6	79.5	68.5	77.2	71.0	78.3
Games	81.1	79.7	85.0	79.1	83.0	79.4
Health	68.7	87.3	65.9	86.3	67.3	86.8
News	84.6	83.6	83.2	78.7	83.9	81.1
Science	74.1	78.9	70.4	83.2	72.2	81.0
Shop	47.2	86.3	39.6	88	43.1	87.1
Society	72.2	82.9	76.1	56.9	74.1	67.5
Sports	86.8	82.7	87.9	84.4	87.3	83.5

Table 6: SVM classification performance of each category on ODP and ACC

Category	Accuracy	Recall	F1
Arts	63.2%	55.5%	59.1%
Business	34.3%	54.6%	42.1%
Computers	57.7%	61.9%	59.7%
Games	37.4%	77.3%	50.4%
Health	22.4%	81.4%	35.1%
News	21.4%	68.9%	32.7%
Science	40.8%	54.5%	46.7%
Shopping	37.0%	17.4%	23.7%
Society	35.2%	13.5%	13.5%
Sports	53.2%	78.7%	63.5%

Table 7: SVM Cross-Test Result

5 Conclusion

In this paper, we proposed a new automatic approach which makes use of web resources to construct the corpora. An automatic system for constructing classification corpora is built. Experiments conducted on ACC and ODP show that the automatic corpora construction approach is effective, although the cross-test result is not satisfactory. Future research will focus on solving the coverage problem of ACC.

Acknowledgment

This work is supported in part by National Natural Science Foundation of China (No.61173075 and No.61272383).

References

Alfio Gliozzo, Carlo Strapparava and Ido Dagan. 2009 *Improving text categorization bootstrapping via unsupervised learning*. ACM Transactions on Speech and Language Processing.

Bart Jongejan and Hercules Dalianis. 2009. *Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike*. Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP, pp.145–153.

Chen-Ming Hung and Lee-Feng Chien. 1999. *Text classification using web corpora and em algorithms*. The Asia Information Retrieval Symposium, pp.12–23.

Chien-Chung Huang, Kuan-Ming Lin and Lee-Feng Chien. 2005. *Automatic training corpora acquisition through web mining*. IEEE/WIC/ACM Conference on Web Intelligence.

Chien-Chung Huang, Shui-Lung Chuang and Lee-Feng Chien. 2004. *Liveclassifier:creating hierarchical text classifier through web corpora*. ACM Transactions on Speech and Language Processing.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Eric P. Jiang. 2009. *Semi-supervised text classification using rbf networks*. International Development Association, pp.95–106.

Fabrizio Sebastiani. 2002. *Machine learning in automated text categorization*. CM Computing Surveys, 34(1):1–47.

Fei Sun, Dandan Song and Lejian Liao. 2011. *Dom based content extraction via text density*. SIGIR, pp.245–254.

Jiaxun Wang and Chunping Li. 2011. *An iterative voting method based on word density for text classification*. WIMS.

Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun and Tom mitchell. 2000. *Text classification from labeled and unlabeled documents using em*. Machine Learning, 39(2-3):103–134.

Kjersti Aas and Line Eikvil. 1999. *A Survey*. Text Categorization.

Matthias Keller and Martin Nussbaumer. 2012. *MenuMiner: revealing the information architecture of large web sites by analyzing maximal cliques*. Proceedings of the 21st international conference companion on World Wide Web, ACM: Lyon, France.

Pu-Jen Cheng. 2009 *Web-based unsupervised text classification*.

Ronen Feldman and James Sanger. 2007. *Advanced Approaches in Analyzing Unstructured Data*. The Text Mining Handbook, Cambrige University.

Si Chen, Gongde Guo and Lifei Chen. 2009. *Semi-supervised classification based on clustering ensembles*. Artificial Intelligence and Computational Intelligence.

S. Sathiya Keerthi and Chih-Jen Lin. 2003. *Asymptotic behaviours of support vector machines with gaussian kernel*. MIT Press, Cambrige University.

Tim Weninger, William H. Hsu and Jiawei Han. 2010. *content extraction via tag ratios*. WWW, pp.971–980.

Wei-Yen Day, Chun-Yi Chi, Ruey-Cheng Chen, Pu-Jen Cheng and Pei-Sen Liu. 2009. *Web mining for unsupervised classification*. pp.53–67.

Yiming Yang and Jan O. Pedersen. 1997. *A comparative study on feature selection in text categorization*. International Conference on Machine Learning, pp.412–420.

Youngjoong Ko and Jungyun Seo. 2009. *Text classification from unlabeled documents with bootstrapping and feature projection techniques*. Information Processing and Management, 45(1):70–83.

Yun-Qian Miao and Mohamed Kamel. 2011. *Pairwise optimized rocchio algorithm for text categorization*. Pattern Recognition Letters, 33(2):375–382.