Fully Automatic Text Categorization by Exploiting WordNet

Jianqiang Li, Yu Zhao, and Bo Liu

NEC Laboratories China
11F, Bldg.A, Innovation Plaza, Tsinghua Science Park
Haidian District, Beijing 100084, China
{lijianqiang,zhaoyu,liubo}@research.nec.com.cn

Abstract. This paper proposes a Fully Automatic Categorization approach for Text (FACT) by exploiting the semantic features from WordNet and document clustering. In FACT, the training data is constructed automatically by using the knowledge of the category name. With the support of WordNet, it first uses the category name to generate a set of features for the corresponding category. Then, a set of documents is labeled according to such features. To reduce the possible bias originating from the category name and generated features, document clustering is used to refine the quality of initial labeling. The training data are subsequently constructed to train the discriminative classifier. The empirical experiments show that the best performance of FACT can achieve more than 90% of the baseline SVM classifiers in F1 measure, which demonstrates the effectiveness of the proposed approach.

Keywords: WordNet, Text Categorization, Semantics.

1 Introduction

Supervised learning is the dominant approach for Text Categorization (TC) [15] [21]. Its performance depends heavily on the quantity and quality of hand-labeled documents. To reduce the burden of manual labeling, semi-supervised approaches [19] use the knowledge from both labeled and unlabeled data for classifier training.

This paper proposes FACT approach. Its underlying assumption is that the category name is specified in human-understandable words, which is generally true for many real applications with good human-computer interfaces. FACT employs the semantic of the category name and the hidden knowledge of the document set for automatic training data construction. First, the category name is extended as a set of representative keywords for each category, which serves as the Representative Profile (RP) of the category to initially label a set of documents. Second, the document clustering is used to refine the initial document labeling, which acts as a regulator to reduce the possible bias derived from the category name. Finally, the training data is constructed from the labeled documents. They are used to supervise the classifier learning.

The key of FACT approach is automatic document labeling. Its basic idea derived from following observation: Given the category name, the prerequisite for the human experts to manually label the documents is that they know the meanings of the

G.G. Lee et al. (Eds.): AIRS 2009, LNCS 5839, pp. 1–12, 2009.

concepts implied by the category name. With the knowledge in mind to read the documents, the expert can assign the category label on them. The knowledge stored in their memories serves as the intermediate to link the categories and documents together. Similarly, with the availability of lexical databases such as WordNet, EuroWordNet, CoreNet, and HowNet, an intuitive way to simulate human's document labeling is to use these lexical resources to provide the same functionality as that of human's knowledge in their brains. In FACT, the description on the concepts implied in the category name and their definitions in WordNet are used as a bridge to provide the linkage between the category and the unlabeled documents.

2 Related Work

Both supervised [15] and semi-supervised [3] [8] [11] TC methods treat category name only as symbolic labels that assume no additional knowledge about them available to help building the classifier. So more or less, certain amount of manual data labeling is required. Our FACT is different. The semantics of the words appeared in the category name is used to supervise the classifier learning. Then, no manual labeling efforts is required in FACT.

FACT is closely related to unsupervised TC approaches. Without labeled documents, they utilize category names [1] or user-specified keywords [20] as the RP for training data building. Since it was hard for users to provide such keywords, [2] uses document clustering together with feature selection to find a set of important words to assist users for keyword selection. FACT is different since it utilizes WordNet to automatically generate a set of representative words as the extended features of the category.

Our work also relates to applying WordNet for automatic TC. [4] proposed to utilize the synonyms in WordNet to improve TC. Several similar techniques are reported to incorporate the synonyms [9], hypernyms [12], hyponyms [13], meronyms and holonyms [16] of words found in the training documents for classifier training. These researches mainly focus on incorporating WordNet to improve the TC model, where the labeled documents are still used. They are different from FACT since we need no labeled data.

3 Our FACT Approach

Given a set of categories C and a set of unlabeled documents D, our FACT consists of four steps: (1) Initial document labeling; (2) Refinement of the initial document labeling; (3) Training data construction; (4) Classifier building. Figure 1 is the flow chart of FACT.

3.1 Initial Document Labeling

Category names are used to initially label some documents in D. It includes three sub-steps: 1) Category name understanding; 2) RP generation; 3) Initial document labeling.

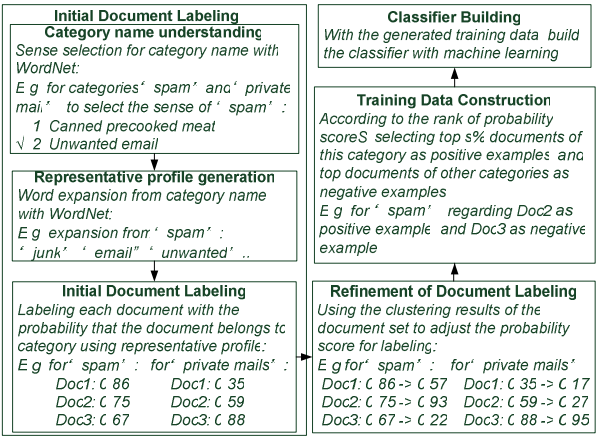

Fig. 1. The flow chart of FACT

1) Category Name Understanding

The goal of this sub-step is to find the relevant word senses of the word/phrase appeared in the category name. For each category name, a preprocessing is conducted. The words inside are tagged for their POS. Here, the word also refers to the simple phrase that can be found in WordNet. After the stop-words (conjunctions, prepositions, and pronouns) are eliminated, the remaining words are used to represent the concepts of the category. They will serve as the seed features to be extended as the RP of corresponding category.

WordNet organizes English nouns, verbs, adjectives, and adverbs into synonym sets, called synsets. Different relationships are defined to link these synsets. One simple method to extend the seed features is just to find all the synsets containing the seed features, then utilize the defined semantic relations to collect the semantically relevant words as the RP of the category. However, there are generally multiple senses (synsets) for each seed word. The homonyms or polysemes could introduce potential noise. So a sense ranking algorithm is given to determine which senses should be considered.

Similar to existing WSD methods [7] using the surrounding words in the sentence to select the correct sense, we propose a sense ranking approach for a word w in the name of category $c_j \in C$ by using the contextual words appeared in the names of category c_j (if the category name contains multiple words) and c_j's adjacent categories (i.e., its sibling, children, and parent categories defined in C, if there is a hierarchical structure inside). For the example given in Figure 1, *spam* and *private emails* are two categories. The words "private" and "email" is the contextual words of "spam".

Each word sense in WordNet is represented as a synset. It has a gloss that provides a linguistic micro-context for that sense. The senses of word w are ranked according to the relatedness of their linguistic micro-contexts with the contextual words of w found in the category names in C.

For a word w, assuming it has m senses $\{w^1, w^2, \ldots, w^m\}$, there are n contextual words $\{cw_1, cw_2, \ldots, cw_n\}$. Each word cw_i has m_i senses $\{cw_i^1, cw_i^2, ..., cw_i^{m_i}\}$, $1 \le i \le n$, the word sense ranks are calculated as follows:

1. For each word sense w^r, $1 \le r \le m$, its relatedness with cw_i^k, i.e., $R(w^r, cw_i^k)$, $1 \le k \le m_i$, is computed. Its value is determined by the cosine of the angle between their gloss vectors. Due to the possible data sparseness caused by the extremely short glosses, this measure is also augmented by the glosses of adjacent hypernyms and hyponyms.
2. Determine the relatedness between a word sense w^r and word cw_i by the sum of $R(w^r, cw_i^k)$, $1 \le k \le m_i$.
3. For each sense w^r, its rank is calculated: $Rank(w^r) = \sum_{1 \le i \le n} \sum_{k=1}^{m_j} f_r \times R(w^r, cw_i^k)$

where each sense of the contextual word is weighted by its frequency count f_r to indicate how important the word sense is.

The rank value reflects the extent to which this word sense shares information in common with its contextual words. The real meaning of the words in a category name might cover several senses, i.e., several senses defined in WordNet might be relevant to the concepts implied by the category name. So, different from traditional WSD that selects only one correct sense, we here need to choose several of them. Intuitively, we can select the top $t\%$ senses as the target senses for feature extension. However, the threshold value is difficult to tune. To handle such a problem, we use a compromised policy, i.e., the higher value of m, the lower value of t is set.

2) RP Generation

Based on the selected word senses, WordNet is used to construct a set of extended keywords for each category, which will serve as the RP rp_j of corresponding category $c_j \in C$. The basic idea is to use the multiple relations defined in WordNet to extract the semantically related words as extended features for each category.

For the category name with only one word: The rp_j is constituted by all the synonyms, hypernyms, hyponyms, meronyms, and holonyms that are identified using the selected senses and the semantic relations defined in WordNet. Also, the words from the derivationally related forms defined in WordNet for corresponding synsets are also used as the representative keywords.

For the category name with multiple words: Besides the keywords found by the multiple semantic relations in WordNet, new phrases are also constructed as a part of the rp_j by automatic synonym substitution. It means that the synonyms are utilized to substitute corresponding word to construct a new phrase. For example, a category name is "spam detection", the synonym of "spam" is "junk e-mail", then "junk e-mail detection" is also selected as the extended features of corresponding category.

Considering that the more relevant sense should contribute more to the TC model, we use the ranking value of each sense to assign the weight to the extended features from this sense, i.e., the word sense with higher rank value will play a more important role in identifying the relevant documents as training data. Intuitively, a

keyword might be more discriminative for one class than the others. However, the experiments show that the TC performance of the final classifier is not sensitive to the weighting policies.

3) Initial Document Labeling

Once the *rp* for each category is constructed, we then apply it for probabilistic document labeling. The probabilistic labeling is based on the similarity of each document $d_i \in D$ with each rp_j. We use the popular cosine similarity metric to measure the similarity. It is based on the Vector Space Model (VSM) of the representations of the rp_j and d_i. $d_i \in D$ is compared with rp_j of category $c_j \in C$ using the cosine metric, and a score $s(d_i,c_j)$ is obtained to indicate similarity between d_i and rp_j.

Using the similarity scores between categories and documents, we can automatically generate a set of probabilistically labeled documents from *D* for corresponding category.

The initial probability value reflects to what extend a document belongs to the category. It is generated only based on the similarity sores between the document and the RP of corresponding category. There are many cases that one document has multiple probability values regarding to different categories. So, for each $c_j \in C$, assuming mx_j is the maximum $s(d_k,c_j)$, $p(c_j|s(d_i,c_j)) = s(d_i,c_j)/mx_j$, we use following formula to normalize the possibility value across multi-categories,

$$P(c_j \mid d_i) = \frac{P(c_j \mid s(d_i, c_j))}{\sum_{c \in C} P(c_j \mid s(d_i, c))}$$

3.2 Refinement of the Initial Document Labeling

Since the initial document labeling might be biased by the RPs, this step uses the document clustering to adjust the initial probability value of the label.

The adjustment is conducted by an alignment model based on the Bayesian inference. Assuming C' be the resultant cluster set and document d_i is clustered into cluster $c'_k \in C'$, based on the Bayesian probability, we have

$$P(c_j \mid d_i, c'_k) = \frac{P(c_j \mid d_i) P(c'_k \mid c_j)}{P(c'_k)},$$

where $P(c_j|d_i)$ can be obtained from the probabilistic document labeling, and the other two components could be obtained with following two equations:

$$P(c'_k \mid c_j) = \frac{\sum_{d \in c'_k} P(c_j \mid d)}{\sum_{d \in D} P(c_j \mid d)}, \quad P(c'_k) = \sum_{c \in C} P(c'_k \mid c),$$

3.3 Training Data Construction and Classifier Building

For each category, there is a list of documents with $P(c_j|d_i, c'_k) \geq 0$. A document could be labeled by multiple categories with certain probabilities. With these multi-labeled documents, the training data are generated automatically.

For each category, the policy for training data construction for category c_j can be described as: 1) The top p^+% documents of the list are selected as positive samples for c_j; 2) The documents at the top p^-% of the list from other categories are identified as reliable negative samples of c_j. Basically, higher $p^+(p^-)$ means more resultant training data but with lower quality; and lower $p^+(p^-)$ indicates less resultant training data but with higher quality. They need to be tuned carefully for the final classifier building.

There might be that multiple categories share the same documents as positive samples. To maximize the discriminative power of the training data, such shared documents only serve as the positive samples for the category with highest probability value.

With the constructed training data, classifiers are built by two discriminative methods, i.e., SVM [17] and TSVM [8].

4 Experiment and Evaluation

1) Experiments Setup

Three English datasets, i.e, 20-NewsGroup (20NP), Reuters-21578, and WebKB, were used. WebKB data set consists of a collection of web pages gathered from university computer science departments, which are divided into seven categories. Our experiment uses four most populous categories: student, faculty, course and project—all together containing 4199 pages. Reuters-21578 is a collection of documents from the Reuters. As many researches [8][15] were conducted, only the most populous 10 classes from the whole dataset are used for our experiment, i.e., Earn, Acquisition, Grain, Crude, Trade, Interest, Ship, Wheat and Corn. There are totally 9296 documents covered by these classes. For each category, the ModApte split is adopted to obtain the training and test data. 20NP is a collection of approximately 20000 articles from 20 different UseNet discussion groups. There are 1000 news documents for each NP. For simplicity, we select two of the 4 main categories, i.e., *Science* (SCI), *Computing* (COMP), as the representation of this dataset. There are totally 9 subcategories, 4 in *Science* and 5 in *Computing*. Since we assume the category name should be specified as normal words in natural language, the abbreviation of the category name is transformed into its dictionary form, e.g., science for SCI, computing for COMP. For baseline approaches, the given training and test sets are used.

For each document, the title and body are extracted as a single feature vector. After the stop-words elimination and stemming, we select the traditional *tf-idf* term weighting scheme as our document representation model. In our experiment, the SVM-light package [8] is employed for the implementation of SVM and TSVM, where a linear kernel is used, and the weight C of the slack variables is set to default.

2) Evaluation Criteria

We use the three standard criteria for binary TC classifier, precision, recall and F1 measure. For the multi-classification problem, we adopt both document-centric microaveraging F1 measure and category-centric macroaveraging F1. The accuracy, i.e., the proportion of documents with correct labels, is adopted to measure the quality of document labeling and the used training data.

3) Experiment Results

We conduct several experiments to tune the parameters of our algorithm.

Category name understanding: The compromised policy for selecting the top t% from the ranking result of m senses is given in Figure 2.

```
if mi≥15,
    the top 35% of the mi senses are selected; {at least 5 senses are selected}
else if 10≤mi<15,
    the top 40% of the mi senses are selected; {at least 4 senses are selected}
else if 6≤mi<10,
    the top 50% of the mi senses are selected; {at least 3 senses are selected}
else
    the top 60% of the mi senses are selected; {for mi=2, if the two synsets
    share same words which are different from the target word, both senses
    are selected}
```

Fig. 2. The policy for word sense selection

Actually, several experiments on word sense selection have been conducted to tune the settings of m and t. The results show that slight differences on the parameters defined in Fig. 2 have almost no effect on the final classification results.

Initial document labeling: We change the percentage pc% ranging from 10-100% to draw the curves of the quality of document labeling. The solid lines in Figure 3 are the result. In Figure 3, we find that the initial document labeling for the Reuters-21578 has the best quality, and WebKB is the poorest dataset. It indicates that, for Reuters-21578, its category names have the best quality to represent the content of its documents. However, the category names used in WebKB don't reflect appropriately its document contents. Also, we observe that, for Reuters-21578 and 20NP, the qualities of the document labeling decrease monotonously with pc. It is consistent with the initial assumption that smaller pc means more reliable labeled data. But for WebKB, after pc>0.2, the accuracy of the labeled document increases with pc. It is due to that there are biased keywords in the RPs.

Refinement of the probabilistic labeling: The k-means algorithm is adopted for document clustering. We use Weka [6] for its implementation. The double of the actual

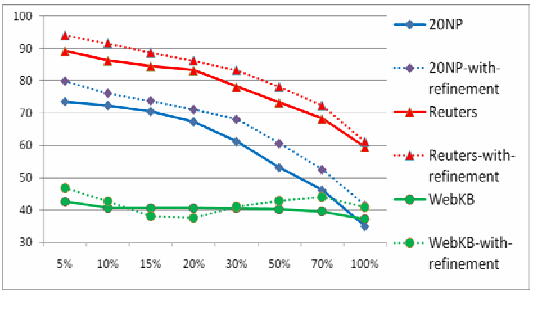

Fig. 3. Document labeling and refinement

number of classes of each dataset is used to set the value of k. The dotted lines in Figure 3 are the accuracies of the labeled documents after the refinement step. Averagely, there is 6-8% improvement. It shows the usefulness of our method for labeled data refinement.

The value setting of k does not matter much for FACT as long as it is not too small (generally, the value of k should be bigger than the actual number of the categories). We also experimented with k being the 1.5 and 4 times of the actual class numbers, the results are very similar to that shown in Figure 3.

Training data construction: We investigate the accuracy of the constructed training data when p^+% is set value from 10%-100%. We know that, when p^+ is given, the actual number of positive samples is determined, e.g., s documents are selected as positive samples for category c_j. Then we can decide how many negative samples are used, e.g., the amount of negative samples is a times of s. a is the parameter to reflect the ratio between negative and positive samples. Obviously, there is a bijective relation between a and p^-. For simplicity, in our implementation, we use a to represent the selected p^-.

Figure 4 shows the accuracies of the constructed training data based on different values of p^+ and a. *all* denotes all the positive samples of other categories are selected as negative samples of this category. We can observe that, with a fixed p^+(or a), the accuracies of the training data decrease with the increase of value a(or p^+). It means that, the more negative (or positive) samples is involved in the training data, the lower the quality of the training data is. Since the precision of negative samples is much higher than that of positive ones, the accuracy of training data is much higher than that of labeled data in Figure 3.

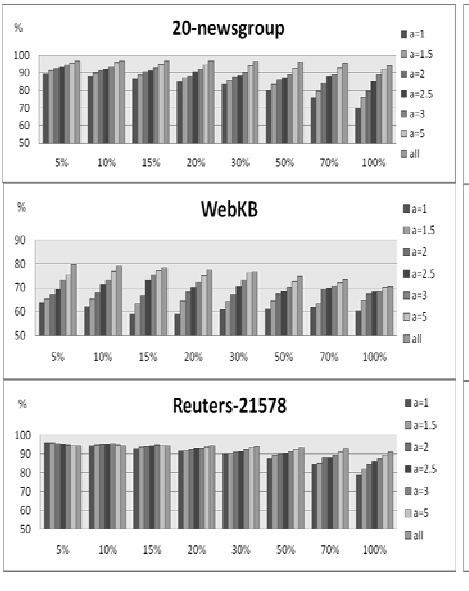

Fig. 4. Accuracy of the used training data

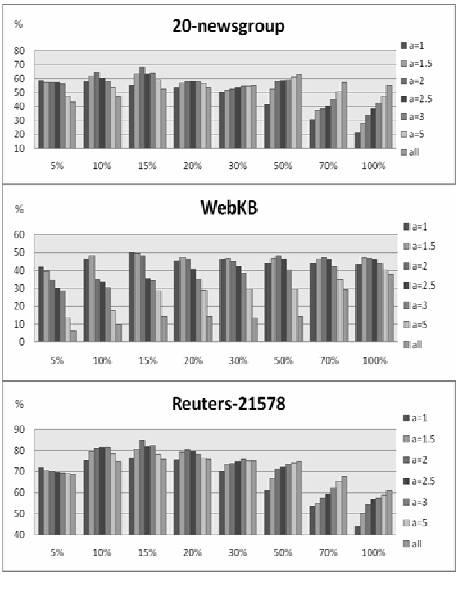

Fig. 5. Selection of parameters p+ and a

Classifier building: In our algorithm, p^+ is set to 15% and a is set to 1.5. The settings are based on empirical experiments which will be explained below.

We select **SVM** and **TSVM** as baseline approaches. For the baseline, the given training and test data from each of the three datasets are used for building and evaluating the classifiers. We implement our FACT approach into two versions, i.e., **SVM+WordNet** and **TSVM+WordNet**. **SVM(TSVM)+WordNet** means that all the documents in the training dataset are treated as unlabeled data to obtain the training data automatically; then SVM (TSVM) classifier is learned; and the test data are used for its evaluation.

Table 1. Comparison of FACT with baselines

Dataset and method \ Measure (%)		Macro-averaging			Micro-averaging		
		Prec	Rec	F1	Prec	Rec	F1
20NP	SVM(baseline)	91.1	65.5	76.3	91.2	63.8	75.1
	TSVM(baseline)	80.7	80.8	80.8	80.7	80.8	80.8
	SVM+WordNet	68.9	66.9	67.9	69.9	67.0	68.4
	TSVM+WordNet	14.6	95.4	25.4	14.6	85.8	25.0
	Without WordNet	26.6	81.0	40.1	26.6	48.6	34.4
Web KB	SVM (baseline)	92.3	74.5	82.5	92.3	72.4	81.2
	TSVM(baseline)	85.2	85.4	85.3	85.2	85.4	85.3
	SVM+WordNet	37.0	72.9	49.1	37.0	70.0	48.4
	TSVM+WordNet	36.8	87.4	51.8	36.8	85.8	51.5
	Without WordNet	27.3	97.9	42.7	27.3	54.5	36.3
Reuters-21778	SVM(baseline)	86.1	97.0	91.2	86.1	96.9	91.2
	TSVM(baseline)	92.7	92.5	92.6	92.7	91.7	92.2
	SVM+WordNet	85.7	84.6	85.1	85.7	83.4	84.5
	TSVM+WordNet	20.7	88.3	33.5	20.7	97.4	34.1
	Without WordNet	53.1	82.2	64.5	53.1	66.3	59.0

Table 1 illustrates the TC results. We observe that, when **SVM+WordNet** is used to categorize the Reuters-21578 and 20NP, FACT can achieve more than 90% of F1 performance of the baseline SVM methods. It proves the effectiveness of FACT.

The unsupervised method given in [1], which uses only the category name to bootstrap a TC classifier, achieves 65% F1 performance of the baseline SVM for 20NP (Although the document set between the two studies for 20NP were not totally the same, the overall ratios of training set to test set were almost exactly the same). The F1 measures are 0.74 for Reuters21578 and 0.65 for 20NP. Since we utilize the semantic knowledge of the category name defined in WordNet (as an external knowledge source) to bridge the categories and documents, as shown in Table 1, the **SVM+WordNet** classifier outperforms it. The TC classifier obtained by labeling words [2] has similar performance comparing to supervised approach. As mentioned in [1], its reason may be the easier TC task and the weaker NB classifier. Actually, the human intervention in selecting important words for each category might result in the high quality of the category's representative words. This fact also makes it possible that their TC results have better quality comparing with our FACT using WordNet to generate the representative words automatically.

For the phenomenon that SVM with inaccurate and insufficient training data could get satisfactory categorization results, we conjecture that the main reason is that the

training data is more compact than the training data used for baseline method. Since we select top ranked documents from the initially labeled documents as training data, the similarity computing between the documents and RPs of the category realizes clustering for the documents in some sense. This can cause that the margin between positive and negative examples much wider than the baseline method. Thus, it might generate a more robust classifier for the document collection. Figure 6 shows this phenomenon.

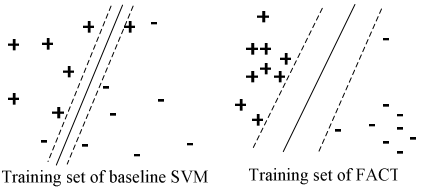

Fig. 6. Baseline SVM and FACT

However, for WebKB, FACT is worse than the baselines. It is because WebKB consists of web pages where many words have no semantic relation with the category name, e.g., Email, date, phone, etc. Then, the limitation of FACT is that it is sensitive not only to the expressiveness of the category name but also to whether the documents are semantically consistent to the category name. But considering the fact that for a real application with good human-computer interface, the category are generally specified clearly in a human-understandable way, it can make the generated features from WordNet and then the resulting training data (i.e., the correctness of the category labels) with good quality.

For the baselines, since the knowledge implied in the unlabeled documents is exploited, TSVM classifier outperforms SVM classifier. However, **SVM+WordNet** outperforms **TSVM+WordNet** a lot on Reuters-21578 and 20NP. This can be explained by that, since TSVM uses both the unlabeled and labeled data for classifier training, then for the resulting training data, the influence of the characteristic that negative and positive documents are separated with a wide margin is decreased a lot comparing that for SVM.

Regarding training data selection from the initial labeled documents, we conduct several experiments to select the value of p^+ and a using SVM. Figure 5 show the results. For 20NP and Reuters-21578, we can see that, when $p^+<0.5$, with the increase of a, the F1 measures increase at the beginning and reach the peak value when a is within 1~3, and then decrease slowly. However, for $p^+\geq 0.5$, the F1 measure increase monotonously with a. We know that, (1) generally, when more training data are utilized, the classifier will get higher accuracy. However, for FACT, (2) when more labeled documents are utilized, more incorrectly labeled documents might be introduced into the training data, which will degrade the categorization results. The above phenomena can be explained by the tradeoff between these two trends of (1) and (2). As $p^+<0.5$, the positive samples have high quality. When a increases at the beginning, since the involved negative samples also with high quality, the trend (1) plays the major role. It causes the enhanced categorization performance with the increase of a.

When a reaches certain value, more noise data from the negative sample is introduced. It causes that trend (2) becomes the principle factor. And then, the performance began to decrease. When $p^+ \geq 0.5$, the low quality of positive samples always make trend (1) play the principle role. The performance of the classifier grows slowly. However, for WebKB, F1 doesn't have the similar behavior. The reason might be that its category name is not semantically consistent with document contents.

Figure 5 shows that, when p^+ is within 0.1~0.2, and a within 1~3, the learned classifier has almost the best performance. It is generally true for the three datasets. Then, in our implementation, we set p^+ and a to the medium values of their best performance ranges.

The contribution of WordNet: To evaluate the impact of WordNet on the results, additional experiments with SVM were done: **Without WordNet**, where the RP only contains the words from category names. The results are also shown in Table 1. Since WordNet bring more knowledge into the classifiers, in all the three datasets, **SVM+WordNet** outperforms **without WordNet** significantly. It demonstrates that using the training data from our automatic labeling approach can provide significant advantage than applying a simple query only consisting in the name of the category. Also, the contribution of WordNet for WebKB is not as notable as for 20NP and Reuters. It might be due to that WebKB's category names are not consistent with its documents, which make the contribution of generated features from WordNet is not stable (although it affect the results obtained without WordNet as well).

5 Conclusion

This paper proposes FACT for fully automatic TC. With the support of WordNet, the semantics of the category name are utilized for automatic document labeling. The document clustering is employed to reduce the possible biases derived from the category name and WordNet. The experiments show that, when the given category name has a clear representation of the topics described in the content of the documents, its performance is very close to the supervised method. Our future work will analyze its statistical significance and extend and evaluate the proposed approach for multilingual TC tasks.

References

1. Gliozzo, A.M., Strapparava, C., Dagan, I.: Investigating Unsupervised Learning for Text Categorization Bootstrapping. In: Proc. of EMNLP (2005)
2. Liu, B., Li, X., Lee, W.S., Yu, P.S.: Text Classification by Labeling Words. In: Proc. 19th Nat'l Conf. Artificial Intelligence (2004)
3. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proc. of the Workshop on Computational Learning Theory (1998)
4. de Buenaga Rodriguez, M., Gomez-Hidalgo, J., Diaz- Agudo, B.: Using WordNet to complement training information in text categorization. In: Proc. of RANLP (1997)
5. Hotho, A., Staab, S., Stumme, G.: Wordnet Improves Text Document Clustering. In: Proc. of the Semantic Web Workshop at SIGIR (2003)

6. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
7. Ide, N., Véronis, J.: Word sense disambiguation: The state of the art. Computational Linguistics 24(1), 1–40 (1998)
8. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proc. 16th International Conf. on Machine Learning, pp. 200–209 (1999)
9. Kehagias, A., Petridis, V., Kaburlasos, V., Fragkou, P.: A comparison of word- and sense-based text classification using several classification algorithms. Journal of Intelligent Information Systems 21(3), 227–247 (2003)
10. Moldovan, D.I., Mihalcea, R.: Using WordNet and Lexical Operators to Improve Internet Searches. IEEE lnternet Computing 4(1), 34–43 (2000)
11. Nigam, K., McCallum, A., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Machine Learning, 103–134 (2000)
12. Scott, S., Matwin, S.: Text classification using WordNet hypernyms. In: Proc. Coling-ACL 1998, pp. 45–52 (1998)
13. Peng, X., Choi, B.: Document classifications based on word semantic hierarchies. In: Proc. of the International Conf. on Artificial Intelligence and Application (AIA 2005), pp. 362–367 (2005)
14. Banerjee, S., Pedersen, T.: An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In: Gelbukh, A. (ed.) CICLing 2002. LNCS, vol. 2276, pp. 136–145. Springer, Heidelberg (2002)
15. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys 34(1), 1–47 (2002)
16. Mansuy, T.N., Hilderman, R.J.: A Characterization of Wordnet Features in Boolean Models For Text Classification. In: AusDM 2006, pp. 103–109 (2006)
17. Vapnik, V.: The nature of statistical learning theory. Springer, Heidelberg (1995)
18. Chen, W., Zhu, J., Wu, H., Yao, T.: Automatic learning features using bootstrapping for text categorization. In: Gelbukh, A. (ed.) CICLing 2004. LNCS, vol. 2945, pp. 571–579. Springer, Heidelberg (2004)
19. Zhu, X.-J.: Semi-Supervised Learning Literature Survey (2007), `http://pages.cs.wisc.edu/~jerryzhu/research/ssl/semireview.html`
20. Ko, Y., Seo, J.: Automatic text categorization by unsupervised learning. In: Proc. of COLING 2000 (2000)
21. Yang, Y., Liu, X.: A re-examination of text categorization methods. In: Proc. of SIGIR 1999 (1999)