An Unsupervised Approach to Domain-Specific Term Extraction

Su Nam Kim[♠], **Timothy Baldwin**[♡]
[♠] [♡] CSSE
[♡] NICTA VRL
University of Melbourne
VIC 3010 Australia
sunamkim@gmail.com, tb@ldwin.net

Min-Yen Kan[♣]
[♣] School of Computing
National University of Singapore
kanmy@comp.nus.edu.sg

Abstract

Domain-specific terms provide vital semantic information for many natural language processing (NLP) tasks and applications, but remain a largely untapped resource in the field. In this paper, we propose an unsupervised method to extract domain-specific terms from the Reuters document collection using term frequency and inverse document frequency.

1 Introduction

Automatic domain-specific term extraction is a categorization/classification task where terms are categorized into a set of predefined domains. It has been employed in tasks such as keyphrase extraction (Frank et al., 1999; Witten et al., 1999), word sense disambiguation (Magnini et al., 2002), and query expansion and cross-lingual text categorization (Rigutini et al., 2005). Even though the approach shows promise, relatively little research has been carried out to study its effects in detail (Drouin, 2004; Milne et al., 2006; Rigutini et al., 2006; Kida et al., 2007; Park et al., 2008). Most of the research to date on domain-specific term extraction has employed supervised machine learning, within the fields of term categorization and text mining. However, to date, the only research to approach the task in an unsupervised manner is that of Park et al. (2008). Unsupervised methods have the obvious advantage that they circumvent the need for laborious manual classification of training instances, and are thus readily applicable to arbitrary sets of domains, tasks and languages.

In this paper, we present a novel unsupervised method for automatically extracting domain-specific terms, targeted specifically at building domain-specific lexicons for natural language processing (NLP) purposes. One of the main properties utilized in this work is *domain specificity*. Our notion of *domain specificity* is based on statistical analysis of word usage, and adopts the simple notions of *term frequency* (*TF*) and *inverse document frequency* (*IDF*) over domains to capture their domain specificity.

2 Unsupervised Domain-Specific Term Extraction

In this section, we elaborate on our proposed method, as well as the benchmark method of Park et al. (2008).

2.1 Proposed Method (D1)

Our proposed unsupervised method is based on *TF-IDF*. The basic underlying idea is that domain-specific terms occur in a particular domain with markedly higher frequency than they do in other domains, similar to term frequency patterns captured by *TF-IDF*.

Hence, we compute *TF-IDF* from TF_{ij}, the term frequency of term i from documents in *domain* j, and IDF_i, the inverse domain frequency. The calculation of TF_{ij} is via:

$$TF_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} \quad (1)$$

where n_{ij} is the number of occurrences of term i in the documents associated with domain j. IDF_i is calculated via:

$$IDF_i = log(\frac{|D|}{|\{d : t_i \in d\}|}) \quad (2)$$

where t_i is the term, and D is the set of all domains.

The final *TF-IDF*$_{ij}$ value of a given term is the simple product of TF_{ij} and IDF_i.

Once the task of scoring terms has been completed, we select those terms which have higher

values than a given threshold. We select the threshold heuristically based on the score distribution, specifically choosing the point at which there is a significant drop in *TF-IDF* scores. That is, when the number of domain-specific terms gained at the current similarity is no more than 20% of the previously-accumulated domain-specific terms, we use that similarity as our threshold.

2.2 Benchmark Method (D2)

We compare our proposed method with the only unsupervised domain-specific term extraction method, i.e. the method of Park et al. (2008). Park *et al.* directly compare term frequencies in documents for a given domain d with term frequencies in the general document collection, based on:

$$domain\_specificity(w) = \frac{\frac{c_d(w)}{N_d}}{\frac{c_g(w)}{N_g}} \quad (3)$$

where $c_d(w)$ and $c_g(w)$ denote the number of occurrences of term w in the domain text and general document collection, respectively. N_d and N_g are the numbers of terms in the domain-specific corpus and in the general corpus, respectively. If term w does not occur in the general corpus, then $c_g(w)$ is set to 1; otherwise it is set to the highest count in the general corpus. In the original work, a one million term corpus from mostly news articles was used as the general corpus. The final score is computed as:

$$Final\ Score = \frac{F + M}{N} \times 100 \quad (4)$$

where N is the number of keywords in the reference data, F is the number of falsely-recognized domain-specific terms (false positives), and M is the number of missed domain-specific terms (false negatives). We avoid computing the final score as shown in (4) since we do not have reference data. Instead, we set a threshold by looking for a significant drop in the score (i.e. the score when the number of newly-extracted terms is less than 20% of the previously-learned terms), as in our approach (D1).

2.3 Collecting Domain-Specific Words

To collect domain-specific words, we used *the modified Lewis split* of the *Reuters document collection*,[1] a dataset which has been extensively used for text categorization, since it contains document-level topics (i.e. domains). In detail, *the modified Lewis split* version of the collection is made up of 90 topics and 3,019 and 7,771 test and training documents, respectively. We extract domain-specific terms from the training documents, and use the 3,019 test articles for text categorization and keyphrase extraction evaluation in Section 3.

Domain	D1	D2	Domain	D1	D2
platinum	132	62	oat	115	49
lumber	77	165	lead	71	105
orange	69	160	hog	61	106
pet-chem	55	246	strategic-metal	50	136
income	49	64	fuel	42	80
alum	37	316	rapeseed	35	13
heat	35	58	tin	33	222
silver	29	99	copper	22	236
wpi	20	87	soy-oil	17	18
zinc	14	50	rubber	13	369
gas	13	122	soy-meal	12	23
meal-feed	12	85			

Table 1: Number of extracted domain-specific terms

After collecting words with the proposed (**D1**) and benchmark (**D2**) methods, we compared them in the form of the ratio of domain-specific terms to the number of domains. Among all the domains present in the corpus, we selected 23 domains which had at least 5 articles in both the test and training data splits, both to manually verify the performance of the two methods, and to utilize the collected domain-specific terms in applications. The total number of terms collected from the 386 selected articles were 1,013 and 2,865, respectively. Table 1 shows the number of domain-specific terms extracted by D1 and the method of D2 over the selected 23 domains. D2 extracts nearly three times more domain-specific terms than D1, but the distribution of terms across domains is relatively well proportioned with D1. This preliminary observation suggests that D1 is more reliable than the benchmark system.

2.4 Human Verification

We manually verified how well our proposed method extracts domain-specific terms. Unlike the method of (Drouin, 2004), where experts

[1] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

scored extracted terms for subtle differences in domain specificity, we opted for a simple annotation process involving non-expert annotators. We asked three human annotators to answer "yes" or "no" when given a term and its domain, as predicted by the two methods. Note that before annotating the actual data set, we trained the human annotators in the context of a pilot annotation test. In our human verification process, we attained an accuracy of 40.59% and 36.59% for D1 and D2, respectively, with initial inter-annotator agreement of 69.61% and 73.04%, respectively. Thus, we cautiously conclude that our proposed method performs better than Park et al. (2008). Note that as the work in Park et al. (2008) was originally developed to extract domain-specific terms for use in correcting domain term errors, the authors did not discuss the performance of domain-specific term extraction in isolation.

We made a few observations during the manual verification process. Despite the strict annotation guidelines (which were further adjusted after the pilot test), the agreement actually dropped between the pilot and the final annotation (especially with one of the annotators, namely A_3). We asked the individual annotators about the ease of the verification procedure and the notion of domain specificity, from their individual perspective. It turned out that although the choice of domain specificity was guided by statistical usage, word senses were involved in the decision process to some degree. Additionally, the annotators commented on the subjectivity of statistical markedness of the terms. The average correlations among two annotators are .738 and .673 for D1 and D2, respectively.

3 Applying Domain-Specific Terms

In this section, we evaluate the utility of the collected domain-specific terms via two tasks: text categorization and keyphrase extraction. Our motivation in selecting these tasks is that domain-specific terms should offer a better representation of document topic than general terms.

3.1 Text Categorization

Automatic text categorization is the task of classifying documents into a set of predefined categories.

Type	F1		F3	
	TF	*TF-IDF*	*TF*	*TF-IDF*
Baseline	.473	.660	.477	.677
Domain	.536	.587	–	–
Combined	.583	**.681**	.579	**.681**

Table 2: Performance of text categorization

To build a text categorization model, we first preprocess the documents, perform part-of-speech (POS) tagging using a probabilistic POS tagger,[2] and lemmatization using `morpha` (Minnen et al., 2001). We then build an SVM-based classifier (Joachims, 1998).[3] We use *TF-IDF* for feature weighting, and all unigram terms. Note that when domain-specific terms are multiword noun phrases (NP), we break them down into unigrams based on the findings of Hulth and Megyesi (2006). As baselines, we built systems using unigram terms which occur above a threshold frequency (i.e. frequency $\geq$ 1, 2 and 3 or F1, F2, F3 in Table 2) after removing stop words. Table 2 shows the micro-averaged F-scores of the text categorization task. Note that since using F2 results in the lowest performance, we report only results over thresholds F1 and F3.

Table 2 shows that domain-specific terms alone do not perform well, since only a relatively small volume of domain-specific indexing terms are extracted, compared to the number of unigram terms. However, when combined with a unigram model, they aid unigram models to improve the overall performance. Despite only showing a small improvement, given the relatively small number of domain-specific terms extracted by our method, we confirm that domain-specific terms are useful for categorizing (monolingual) texts, just as domain specificity has been shown to help in cross-lingual text categorization (Rigutini et al., 2005).

3.2 Automatic Keyphrase Extraction

Keyphrases are simplex nouns or NPs that represent the key ideas of the document. They can serve as a representative summary of the document and also as high-quality index terms. In the past, various attempts have been made

[2] `Lingua::EN::Tagger`

[3] `http://svmlight.joachims.org/svm_multiclass.html`

Type	L	*Boolean*	*TF*	*TF-IDF*
KEA	NB	.200	–	–
	ME	.249	–	–
KEA +	NB	.204	.200	.197
Domain	ME	.260	.261	**.267**

Table 3: Performance of keyphrase extraction

to boost automatic keyphrase extraction performance, based primarily on statistics (Frank et al., 1999; Witten et al., 1999) and a rich set of heuristic features (Nguyen and Kan, 2007).

To collect the gold-standard keyphrases, we hired two human annotators to manually assign keyphrases to 210 test articles in the same 23 selected domains. In summary, we collected a total of 1,339 keyphrases containing 911 simplex keyphrases and 428 NPs. We checked the keyphrases found after applying the candidate selection method employed from Nguyen and Kan (2007). The final number of keyphrases found in our data was only 750 (56.01% of all the documents), among which 158 (21.07%) were NPs.

To build a keyphrase extractor, we first preprocessed them with a POS tagger and lemmatizer, and applied the candidate selection method in Nguyen and Kan (2007) to extract candidates. Then, we adopted two features from KEA (Frank et al., 1999; Witten et al., 1999), as well as the domain-specific terms collected by our method. KEA uses two commonly used features: *TF-IDF* for document cohesion, and *distance* to model the locality of keyphrases. Finally, we used the features to build a maxent classifier[4] and a Naïve Bayes (NB) model. To represent the domain specificity of the keyphrase candidates, we simply presented the 23 domains as three separate sets of features with differing values (*Boolean*, *TF* and *TF-IDF*), when a given keyphrase candidate is indeed a domain-specific term. Finally, with KEA as a baseline, we compared the systems over the top-7 candidates using the current standard evaluation method (i.e. *exact matching scheme*). Table 3 shows the micro-averaged F-scores.

In the results, we first notice that our test system outperformed KEA with ME, but that our test system using *Boolean* produced better performance than KEA only with NB. The maximum improvement in F-score is about 1.8%, in the best configuration where *TF-IDF* weighting is used in conjunction with an ME learner. This is particularly notable because: (a) the average performance of current keyphrase extraction systems is a little more than 3 matching keyphrases over the top 15 candidates, but we produce only 7 candidates; and (b) the candidate selection method we employed (Nguyen and Kan, 2007) found only 56.01% of keyphrases as candidates. Finally, we note with cautious optimism that domain-specific terms can help in keyphrase extraction, since keyphrases are similar in the same or similar domains.

[4] http://maxent.sourceforge.net/index.html

4 Conclusions

In this work, we have presented an unsupervised method which automatically extracts domain-specific terms based on term and document statistics, using a simple adaptation of *TF-IDF*. We compared our method with the benchmark method of Park et al. (2008) using human judgements. Although our method did not extract a large number of domain-specific terms, the quality of terms is high and well distributed over all domains. In addition, we have confirmed the utility of domain-specific terms in both text categorization and keyphrase extraction tasks. We empirically verified that domain-specific terms are indeed useful in keyphrase extraction, and to a lesser degree, text categorization. Although we could not conclusively prove the higher utility of these terms, there is a strong indicator that they are useful and deserve further analysis. Additionally, given the small number of domain-specific terms we extracted and used, we conclude that they are useful for text categorization.

Acknowledgements

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

K.V. Chandrinos and I. Androutsopoulos and G. Paliouras and C.D. Spyropoulos. Automatic Web

rating: Filtering obscence content on the Web. In *Proceedings of ECRATDL*. 2000, pp. 403–406.

P. Drouin. Detection of Domain Specific Terminology Using Corpora Comparison. In *Proceedings of LREC*. 2004, pp. 79–82.

G. Escudero and L. Marquez and G. Rigau. Boosting applied to word sense disambiguation. In *Proceedings of 11th ECML*. 2000, pp. 129–141.

E. Frank and G.W. Paynter and I. Witten and C. Gutwin and C.G. Nevill-Manning. Domain Specific Keyphrase Extraction. In *Proceedings of 16th IJCAI*. 1999, pp. 668–673.

A. Hulth and B. Megayesi. A Study on Automatically Extracted Keywords in Text Categorization. In *Proceedings of COLING/ACL*. 2006, pp. 537–544.

T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of ECML*. 1998, pp. 137–142.

M. Kida and M. Tonoike and T. Utsuro and S. Sato. Domain Classification of Technical Terms Using the Web. *Systems and Computers*. 2007, 38(14), pp. 2470–2482.

P. Lafon. Sur la variabilite de la frequence des formes dans un corpus. In *MOTS*. 1980, pp. 128–165.

B. Magnini and C. Strapparava and G. Pezzulo and A. Gliozzo. The role of domain information in word sense disambiguation. *NLE*. 2002, 8(4), pp. 359–373.

D. Milne and O. Medelyan and I.H. Witten. Mining Domain-Specific Thesauri from Wikipedia: A case study. In *Proceedings of the International Conference on Web Intelligence*. 2006.

G. Minnen and J. Carroll and D. Pearce. Applied morphological processing of English. *NLE*. 2001, 7(3), pp.207–223.

T.D. Nguyen and M.Y. Kan. Key phrase Extraction in Scientific Publications. In *Proceeding of ICADL*. 2007, pp. 317-326.

Y. Park and S. Patwardhan and K. Visweswariah and S.C. Gates. An Empirical Analysis of Word Error Rate and Keyword Error Rate. In *Proceedings of ICSLP*. 2008.

L. Rigutini and B. Liu and Marco Magnini. An EM based training algorithm for cross-language text categorization. In *Proceedings of WI*. 2005, pp. 529-535.

L. Rigutini and E. Di Iorio and M. Ernandes and M. Maggini. Automatic term categorization by extracting knowledge from the Web. In *Proceedings of 17th ECAI*. 2006.

G. Salton and A. Wong and C.S. Yang. A vector space model for automatic indexing. *Communications of the ACM*. 1975, 18(11), pp. 61–620.

I. Witten and G. Paynter and E. Frank and C. Gutwin and G. Nevill-Manning. KEA:Practical Automatic Key phrase Extraction. In *Proceedings of ACM DL*. 1999, pp. 254–256.