Journal of Computer Science 6 (3): 323-329, 2010
ISSN 1549-3636

Improving Term Extraction Using Particle Swarm Optimization Techniques

Mohammad Syafrullah and Naomie Salim
Faculty of Computer Science and Information Systems,
University Technology Malaysia, 81310, Skudai, Johor, Malaysia

Abstract: Problem statement: Term extraction is one of the layers in the ontology development process which has the task to extract all the terms contained in the input document automatically. The purpose of this process is to generate list of terms that are relevant to the domain of the input document. In the literature there are many approaches, techniques and algorithms used for term extraction where each of approaches, techniques and algorithms has the objective to improve the precision of the extracted terms. **Approach:** We proposed a new approach using particle swarm optimization techniques in order to improve the precision of term extraction results. We choose five features to represent the term score. **Results:** The approach had been applied to the domain of Islamic documents. We compare our term extraction method with TFIDF, Weirdness, GlossaryExtraction and TermExtractor. **Conclusion:** The experimental results showed that our proposed approach achieves better precision than those four algorithms.

Key words: Term extraction, particle swarm optimization, feature selection, text mining

INTRODUCTION

Recently many experiments have been conducted for term extraction task. Literatures provide many examples of term extraction methods. Most of these are based on linguistic method, terminology and NLP method and the others based on statistical/information retrieval method (Cimiano, 2006).

Many linguistic methods use shallow text processing techniques such as tokenizer, Part-Of-Speech (POS) tagger and syntactic analyzer (parser). For example, Text-to-Onto use linguistic method called Saarbrucken Message Extraction System (SMES) in their system architecture to produce list of terms from the input text (Maedche and Staab, 2003). Another system, SVETLAN, use syntactic analyzer Sylex to find list of terms from the input text (Chalendar and Grau, 2000).

In the study of Frantzi *et al*. (2000), statistical measurement of frequency occurrence is used for the automatic extraction of multi-word terms, from English medical corpus. Park *et al*. (2002) and Kozakov *et al*. (2004) introduced term cohesion to calculate the cohesion of the multi-word terms. The measure is proportional to the co-occurrence frequency and the length of the term. Panel and Lin (2001) present a language independent statistical corpus-based term extraction algorithm. In their algorithm, they collect bigram frequencies from a corpus and extract two-word candidates. After collecting features for each two-word candidate, they use mutual information and Log Likelihood Ratio to extend them into multi-word terms.

In statistical method, statistical analysis will be performed on the input and this analysis will identify terms based on the statistical rank. Most of the statistical methods for term extraction are based on information retrieval method for term indexing (Salton and Buckley, 1988; Yates and Neto, 1999). Other methods use the notion of "weirdness" (Ahmad *et al*., 1999), domain pertinence (Navigli and Velardi, 2004; Sclano and Velardi, 2007) and domain specificity (Park *et al*., 2002; Kozakov *et al*., 2004).

Terminology and NLP approach emphasize on the internal analysis for the term extraction within the corpus, while statistical methods rely on the comparison of frequencies between domain specific and general corpora (external analysis).

Related works: Kea is one of the extraction systems which are using statistical method. It uses TFIDF and first occurrence in the document as its features to determine the weight of each keyphrase. Kea's extraction algorithm has two stages, first is training stage which has the task to create a model for identifying keyphrases, using training documents. The second one is extraction stage which will choose

Corresponding Author: Mohammad Syafrullah, Faculty of Computer Science and Information Systems, University Technology Malaysia, 81310, Skudai, Johor, Malaysia

keyphrases from a test document, using the model that has been made in the previous stage (Witten *et al*., 1999).

Turney (2000) treats the problem of keyphrase extraction as supervised learning task. He presented two approaches to the task of learning to extract keyphrases from text. The first approach was to apply the C4.5 and the second one was using genetic algorithm. Turney's program is called Extractor. One form of this extractor is called GenEx, which is use Genitor genetic algorithm to maximize the performance (fitness) on the training process. Genitor is used to tune Extractor, but is no longer needed once the training process is complete.

GlossaryExtraction (Park *et al*. 2002; Kozakov *et al*., 2004) is a GlossaryExtraction tool that uses two features which are domain specificity and term cohesion for calculating the term weight. GlossaryExtraction algorithm has the two important parts which are identification of candidate glossary items and glossary item ranking and selection. After obtaining candidate glossary items, the algorithm will rank them before selecting the final set. In their research, they claim that their method can improve the document-relevancy ranking compared with log likelihood ratio and mutual information.

The term extraction algorithm called Kea++ is the improvement of the original keyphrase extraction algorithm Kea. Medelyan and Witten (2005) called their new approach as index term extraction, because they combine the advantages of both keyphrase extraction and term assignment into a single scheme. Their preliminary evaluations shows that the Kea++ significantly outperforms compared with Kea extraction algorithm.

Another term extraction systems called TermExtractor (Navigli and Velardi, 2004; Sclano and Velardi, 2007), use three features to compute their term weight. Domain pertinence is used to perform a contrastive analysis between domain of interest documents and other domains documents. Domain consensus is used to measure the distribution of terms in a domain of interest, while the definition of lexical cohesion similar to that already introduced in (Park *et al*., 2002; Kozakov *et al*., 2004).

MATERIALS AND METHODS

We propose a new approach of term extraction, which takes into account several kinds of features, including domain relevance, domain consensus, term cohesion, first occurrence and length of noun phrase, to produce a list of terms.

Two steps are employed in our propose approach. First, terms are ranked to emphasize the most relevant from domain of input document; second, the score function is trained by the particle swarm optimization to obtain a suitable combination of feature weights.

Methodology: The goal of term extraction is to generate list of terms that are relevant to the domain of the input domain. Our proposed approach consists of the following steps:

- Read the input document
- Preprocessing step consist of three sub tasks: Syntactic parser does a syntactic analysis on every input sentence from input document and produces a list of syntactic information (Noun Phrase-NP). Stop words should be filtered from each of the list of NP. Finally, the list of NP should be stemmed to produce list of clean NP, as the term candidate
- Each term candidate is associated with vector that contains five features
- The five features are used to calculate the term score and then rank the terms based on their score

Our propose term extraction approach has two stages:

- Training stages: This stage has the task to create a model for identifying terms using training documents. Features are extracted from training documents and used to train the swarm optimization model
- Extraction stages: This stage will choose terms from a test document (this document is different than that were used for training), using the model that has been made in the training stage

Figure 1 shows our proposed term extraction model. Both stages choose a set of term candidate from their input documents and then calculate the values of certain features for each candidate.

Particle swarm optimization: Particle swarm optimization first introduced by Eberhart and Kennedy (1995a; 1995b) and Eberhart and Shi (1998), as an optimization technique based on the movement and intelligence of a swarm. It was inspired by the social behavior and dynamics of movement of birds and fish. PSO uses a number of particles that constitute a swarm moving around in the search space to find the best solution. Each particle is treated as a point in the search space which adjusts its flight according to its own flying experience and other particles flying experience.

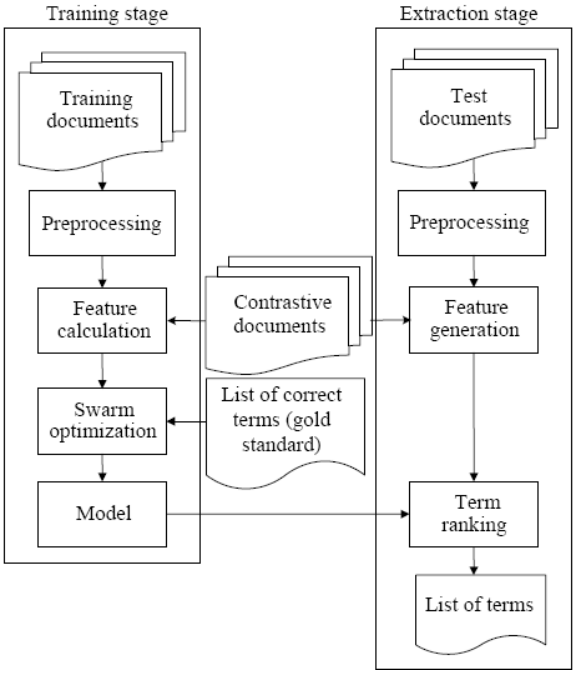

Fig. 1: The training and extraction stage processes

Initially, the PSO algorithm randomly selects candidate solutions within the search space. During each iteration of the algorithm, each particle is evaluated by the objective function being optimized, determining the fitness of the solution. A new velocity value for each particle is calculated using the following equation:

$$v_i(t+1) = wv_i(t) + c_1 r_1 [\hat{x}_i(t) + x_i(t)] + c_2 r_2 [g(t) - x_i(t)] \quad (1)$$

The index of the particle is represented by i. So, $v_i(t)$ is the velocity of particle i at time t and $x_i(t)$ is the position of particle i at time t. Parameters w, c_1 and c_2 are user-supplied coefficients. The values r_1 and r_2 are random values regenerated for each velocity update. Value $\hat{x}_i(t)$ is the individual best candidate solution for particle i at time t and g(t) is the swarm's global best candidate solution at time t. Once the velocity for each particle is calculated, each particle's position is updated by applying the new velocity to the particle's previous position using Eq. 2. This process is then repeated until some stopping condition is met. Figure 2 describes the flowchart of PSO algorithm:

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

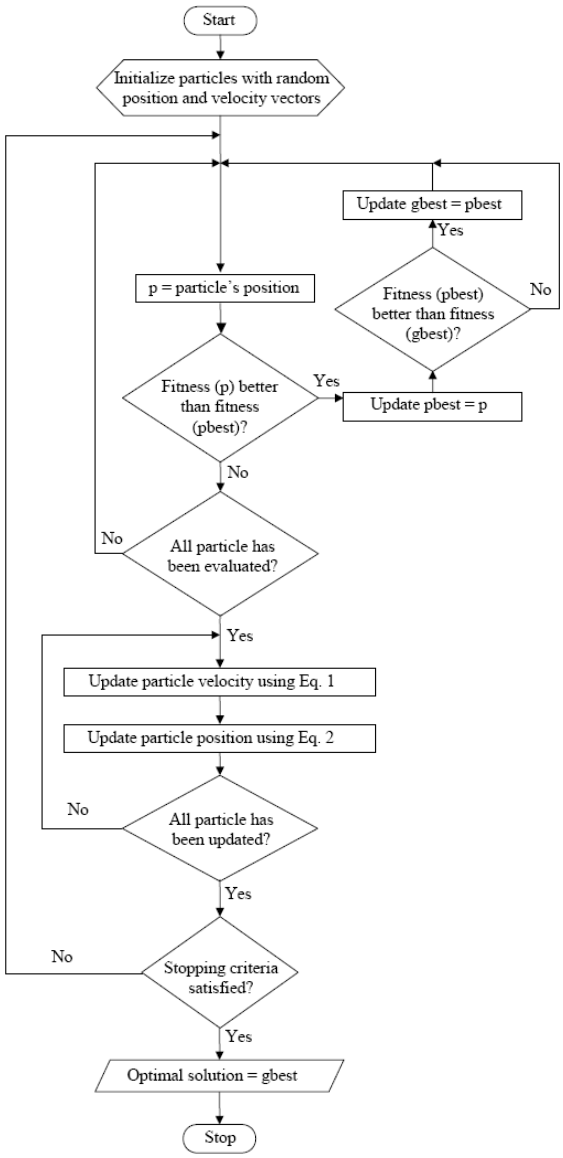

Fig. 2: Flowchart of particle swarm optimization algorithm

Feature definition: In order to characterize the noun phrases in the documents we have adopted five features. These five features are calculated for each candidate term and used both in training and extraction stage. The features used are: domain relevance, domain consensus, term cohesion, first occurrence and length of noun phrase.

f1: Domain relevance-domain relevance can be given according to the amount of information captured in the

target document with respect to contrastive documents. Let D_i is the domain of interest (a set of relevant documents) and $\{D_1...D_n\}$ is sets of documents in another domain, domain relevance of a term t in class D_i is computed as (Navigli and Velardi, 2004; Sclano and Velardi, 2007):

$$DR(t, D_k) = \frac{D(t \mid D_k)}{\max_{1 \le j \le n}(P(t \mid D_j))} \quad (3)$$

where, $(P(t|D_k))$ estimated as:

$$E(P(t \mid D_k)) = \frac{f_{t,k}}{\sum_{t' \in D_k} f_{t',k}} \quad (4)$$

f2: Domain consensus-domain consensus measures the distributed use of a term in a domain D_k. Domain consensus is expressed as follows (Navigli and Velardi, 2004; Sclano and Velardi, 2007):

$$DC(t, D_k) = \sum_{d \in D_k} \left(P(t \mid d).\log_2 \left(\frac{1}{P(t \mid d)} \right) \right) \quad (5)$$

Where:

$$E(P(t \mid d_j) = \frac{f_{t,j}}{\sum_{d_j \in D_k} f_{t,j}} \quad (6)$$

f3: Term cohesion-term cohesion is used to calculate the cohesion of the multi-word terms. The measure is proportional to the co-occurrence frequency and the length of the term (Park *et al*. 2002; Kozakov *et al*., 2004):

$$TC(t) = \frac{|t| .\log_{10}(f(t)).f(t))}{\sum_{w_i \in t} f(w_i)} \quad (7)$$

f4: First occurrence-the main idea behind this feature is that important terms tend to occur at the beginning of documents. First occurrence is calculated as the number of words that precede its first appearance, divided by the number of words in the document. The resulting feature is a number between 0 and 1 representing the proportion of the documents before the term's first appearance (Witten *et al*., 1999; Medelyan and Witten, 2005).

f5: Length of noun phrase-candidate length is also a useful feature in extraction as well as in candidate selection, because the majority of terms are one or two words in length. Length of noun phrase score is calculated as its frequency times its length (in words) (Barker and Corrnacchia, 2000).

Term generation: For a term t, a weighted score function, as shown in the following equation, is used to integrate all the feature scores, where w_i indicates the weight of f_i:

$$Score(t) = \sum_{i=1}^{5} w_i \cdot Score_{f_1}(t) \quad (8)$$

Moreover, the particle swarm optimization is used to obtain an appropriate set of feature weights. We have set the particle swarm optimization variables as follows: Number of particles = 40, maximum number of iterations = 500, $c_1 = 2$, $c_2 = 2$ and w = (0.5+(random/2)). During each iteration of the algorithm, each particle is evaluated using the fitness function as in (9). By applying particle swarm optimization, a suitable combination of feature weights could be found:

$$Fitness = \max\left(\sum_{i=1}^{|extracted|} |t_i \in goldstandard| \right) \quad (9)$$

Where:

|extracted| = A number of terms extracted by the system

$|t_i \in goldstandard|$ = The number of terms that is a member of the gold standard (reference of correct terms)

RESULTS

We use English translation to the meaning of the Quran (focus on verses about prayer) as the input document in the experiment. We separate the documents into a training documents and test documents (4 for training and 1 for testing). In the experiment, we also use Reuters-21578, the documents in the Reuters-21578 collection appeared on the Reuters newswire in 1987. We converted all the documents into 22 plain text file (reut2-000.txt until reut2-021.txt) and use it as contrastive documents. Beside that, we also prepare the gold standard (reference of correct terms) that contain list of the Quran terms (focus on verses about prayer).

Table 1: Term extraction precision for each feature

Precision (feature)	No. of terms 25	50	150	250
f_1	0.800	0.820	0.607	0.552
f_2	0.880	0.760	0.673	0.596
f_3	0.880	0.780	0.673	0.596
f_4	0.800	0.740	0.650	0.610
f_5	0.880	0.740	0.600	0.584

Table 2: Term extraction precision for different number of training/test documents

No. of Train/Test	No. of terms 25	50	150	250
1/1	0.960	0.860	0.673	0.612
2/1	0.920	0.880	0.673	0.612
3/1	0.960	0.860	0.673	0.612
4/1	0.960	0.860	0.673	0.616

Table 3: Weight of each feature for different number of training/test documents

No. of Train/Test	Weight of each feature f1	f2	f3	f4	f5
1/1	0.290	0.453	0.488	0.489	0.433
2/1	0.322	0.475	0.491	0.532	0.364
3/1	0.451	0.481	0.492	0.408	0.361
4/1	0.422	0.495	0.524	0.408	0.460

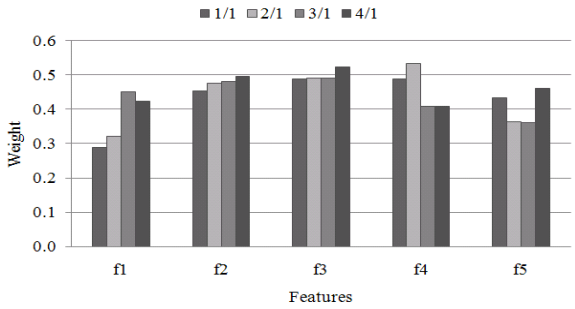

Fig. 3: Weight of each feature for different number of training/test documents

In the extraction stage, we evaluate the precision of our propose methods at 4 points: Top 25, 50, 150 and 250 terms using the following equation:

$$\text{precision} = \frac{\sum_{i=1}^{|\text{extracted}|} | t_i \in \text{golds tan dard} |}{| \text{extracted} |} \quad (10)$$

We compare the terms extracted by the system with the gold standard that we have prepare before. Table 1 shows the term extraction precision for each feature for different number of terms evaluated. Table 2 shows the term extraction precision for different number of training/test documents. Table 3 and Fig. 3 shows the weight of each feature for different number of training/test documents.

Table 4: Comparison of the term extraction precision

Precision (algorithm)	No. of terms 25	50	150	250
TFIDF	0.840	0.800	0.607	0.560
Weirdness	0.760	0.600	0.607	0.588
GlossaryExtraction	0.840	0.740	0.633	0.592
TermExtractor	0.840	0.800	0.647	0.564
Swarm model	0.960	0.860	0.673	0.616

Fig. 4: Comparison of the term extraction precision (Swarm Model, TFIDF, Weirdness, GlossaryExtraction and TermExtractor)

DISCUSSION

We compare the precision of our propose method with four other known algorithms. The result show that our propose method based on particle swarm optimization can improve the precision of the extracted terms. Table 4 and Fig. 4 show the comparison of the precision between swarm model and the four other algorithms (TFIDF, Weirdness, GlossaryExtraction and TermExtractor).

CONCLUSION

We have presented a particle swarm optimization technique to improve term extraction precision. We choose five features to represent the term score: domain relevance, domain consensus, term cohesion, first occurrence and length of noun phrase. In the experiments, we use a translation of the meaning of the Quran (focus on verses of prayer) as an input document, both for training and testing phases. We separate the documents between training documents and test documents. Particles swarm optimization is trained using the training documents to determine the appropriate weight of each feature to produce the best score for each term. We conduct tests with the test document using the weight of each feature which is generated from the training stage to calculate the final score for each term to be extracted. Our experimental results show the use of particle swarm optimization

technique can improve the precision of the extracted terms compared with four other known algorithms (TFIDF, Weirdness, GlossaryExtraction and TermExtractor).

ACKNOWLEDGMENT

This project is sponsored by the Ministry of Science, Technology and Innovation Malaysia under E-Science grant 01-01-06-SF0539.

REFERENCES

Ahmad, K., L. Gillam and L. Tostevin, 1999. University of surrey participation in TREC 8: Weirdness Indexing for Logical Document Extrapolation and Retrieval (WILDER). Proceeding of the 8th Text Retrieval Conference, Nov. 16-19, Department of Commerce, National Institute of Standards and Technology, USA., pp: 717-724. http://trec.nist.gov/pubs/trec8/papers/surrey2.pdf

Barker, K. and N. Corrnacchia, 2000. Using noun phrase heads to extract document keyphrases. Lecturer Notes Comput. Sci., 1822: 40-52. DOI: 10.1007/3-540-45486-1_4

Chalendar, G. and B. Grau, 2000. SVETLAN' or How to Classify Words Using Their Context. In: Knowledge Engineering and Knowledge Management Methods, Models and Tools, Chalendar, G. and B. Grau (Eds.). Springer Berlin/Heidelberg, Germany, ISBN: 978-3-540-41119-2, pp: 99-112.

Cimiano, P., 2006. Ontology Learning and Population from Text: Algorithms, Evaluation and Applications. 1st Edn., Springer, Germany, ISBN: 10: 0-387-30632-3, pp: 3-5.

Eberhart, R.C. and J. Kennedy, 1995a. A new optimizer using particle swarm theory. Proceeding 6th International Symposium on Micro Machine and Human Science, Oct. 04-06, IEEE Service Center, Piscataway, NJ., Nagoya, Japan, pp: 39-43. DOI: 10.1109/MHS.1995.494215

Eberhart, R.C. and J. Kennedy, 1995b. Particle swarm optimization. Proceedings of the IEEE International Conference on Neural Networks, Nov. 27-Dec. 01, IEEE Xplore Press, Perth, Australia, pp: 1942-1948. DOI: 10.1109/ICNN.1995.488968

Eberhart, R.C. and Y.H. Shi, 1998. Comparison between genetic algorithms and particle swarm optimization. Lecture Notes Comput. Sci., 1447: 611-616. DOI: 10.1007/BFb0040812

Frantzi, K., S. Ananiadou and H. Mima, 2000. Automatic recognition of multi-word terms: The C-value/NC-value method. Int. J. Digital Libraries, 3: 115-130. DOI: 10.1007/s007999900023

Kozakov, L., Y. Park, T. Fin, Y. Drissi, Y. Doganata and T. Cofino, 2004. GlossaryExtraction and utilization in the information search and delivery system for IBM technical support. IBM Syst. J., 43: 546-563. http://www.research.ibm.com/people/y/yurdaer/papers/ibmsysjournal2004a.pdf

Maedche, A. and S. Staab, 2003. Mining Ontologies from Text. In: Knowledge Engineering and Knowledge Management Methods, Models and Tools, Maedche, A. and S. Staab (Eds.). Springer Berlin/Heidelberg, Germany, ISBN: 978-3-540-41119-2, pp: 169-189.

Medelyan, O. and I.H. Witten, 2005. Thesaurus-based index term extraction for agricultural documents. Proceeding of the 6th Workshop on Agricultural Ontology Service, European Federation for Information Technology in Agriculture (EFITA) and WCCA (World Congress on Computers in Agriculture), July, Vila Real, Portugal, pp: 1-8. http://www.cs.waikato.ac.nz/~olena/publications/efita2005_kea.pdf

Navigli, R. and P. Velardi, 2004. Learning domain ontologies from document warehouses and dedicated web sites. Comput. Linguist., 30: 151-170. DOI: 10.1162/089120104323093276

Panel, P. and D. Lin, 2001. A statistical corpus-based TermExtractor. Lecture Notes Artif. Intell., 2056: 36-46. DOI: 10.1007/3-540-45153-6_4

Park, Y., R. J. Byrd, and B. K. Boguraev, 2002. Automatic glossaryextraction: Beyond terminology identification. Proceedings of the 19th International Conference on Computational Linguistics, Aug. 26-30, Association for Computational Linguistics (ACL), Taipei, Taiwan, pp: 772-778. http://www.aclweb.org/anthology/C/C02/C02-1142.pdf

Salton, G. and C. Buckley, 1988. Term-weighting approaches in automatic text retrieval. Inform. Process. Manage., 24: 515-523. DOI: 10.1016/0306-4573(88)90021-0

Sclano, F. and P. Velardi, 2007. Termextractor: a web application to learn the shared terminology of emergent web communities. Proceedings of the 3rd International Conference on Interoperability for Enterprise Software and Applications, Mar. 28-30, Springer, Funchal, Portugal, pp: 287-298. DOI: 10.1007/978-1-84628-858-6_32

Turney, P.D, 2000. Learning algorithms for keyphrase extraction. J. Inform. Retriev., 2: 303-336. DOI: 10.1023/A: 1009976227802

Witten, I.H, G.W. Paynter, E. Frank, C. Gutwin and C.G. Nevill-Manning, 1999. KEA: Practical automatic keyphrase extraction. Proceedings of the 4th ACM Conference on Digital Libraries, Aug. 11-14, ACM Publisher, Berkeley, California, USA., pp: 254-256. DOI: 10.1145/313238.313437

Yates, B. and R. Neto, 1999. Modern Information Retrieval. 1st Edn., Addison-Wesley, USA., ISBN: 0-201-39829-X, pp: 544.