Low-Bias Extraction of Domain-Specific Concepts

Axel-Cyrille Ngonga Ngomo
University of Leipzig, Johannisgasse 26, Leipzig D-04103, Germany
E-mail: ngonga@informatik.uni-leipzig.de,
WWW home page: http://bis.uni-leipzig.de/AxelNgonga

Keywords: natural language processing, local graph clustering, knowledge-free concept extraction

Received: February 5, 2009

The availability of domain-specific knowledge models in various forms has led to the development of several tools and applications specialized on complex domains such as bio-medecine, tourism and chemistry. Yet, most of the current approaches to the extraction of domain-specific knowledge from text are limited in their portability to other domains and languages. In this paper, we present and evaluate an approach to the low-bias extraction of domain-specific concepts. Our approach is based on graph clustering and makes no use of a-priori knowledge about the language or the domain to process. Therefore, it can be used on virtually any language. The evaluation is carried out on two data sets of different cleanness and size.

Povzetek: Od jezika neodvisna metoda iz besedila izlušči termine in nato domensko odvisne koncepte.

1 Introduction

The recent availability of domain-specific knowledge models in various forms has led to the development of information systems specialized on complex domains such as bio-medecine, tourism and chemistry. Domain-specific information systems rely on domain knowledge in forms such as terminologies, taxonomies and ontologies to represent, analyze, structure and retrieve information. While this integrated knowledge boosts the accuracy of domain-specific information systems, modeling domain-specific knowledge manually remains a challenging task. Therefore, considerable effort is being invested in developing techniques for the extraction of domain-specific knowledge from various resources in a semi-automatic fashion. Domain-specific text corpora are widely used for this purpose. Yet, most of the current approaches to the extraction of domain-specific knowledge in the form of terminologies or ontologies are limited in their portability to other domains and languages. The limitations result from the knowledge-rich paradigm followed by these approaches, i.e., from them demanding hand-crafted domain-specific and language-specific knowledge as input. Due to these constraints, domain-specific information systems exist currently for a limited number of domains and languages for which domain-specific knowledge models are available. An approach to remedy the high human costs linked with the modeling of domain-specific knowledge is the use of low-bias, i.e., knowledge-poor and unsupervised approaches. They require little human effort but more computational power to achieve the same goals as their hand-crafted counterparts.

In this work, we propose the use of low-bias approaches for the extraction of domain-specific terminology and concepts from text. Especially, we study the low-bias extraction of concepts out of text using a combination of metrics for domain-specific multi-word units and graph clustering techniques. The input for this approach consists exclusively of a domain-specific text corpus. We use the Smoothed Relative Expectation [9] to extract domain-specific multi-word units from the input data set. Subsequently we use SIGNUM [10] to compute a domain-specific lexicon. Finally, we use BorderFlow, a novel general-purpose graph clustering algorithm, to cluster the domain-specific terminologies to concepts. Our approach is unsupervised and makes no use of a-priori knowledge about language-specific patterns. Therefore, it can be applied to virtually all domains and languages. We evaluate our approach on two domain-specific data sets from the bio-medical domain. To achieve this goal, we present both a quantitative evaluation against kNN [19] and a qualitative evaluation against the MEdical Subject Headings(MESH)[1].

The remainder of this paper is structured as follows: first, we present related work on concept extraction. Then, we present our approach to the low-bias extraction of concepts using graph clustering, focusing especially on our clustering technique. Subsequently, we evaluate our concept extraction approach quantitatively and qualitatively. We conclude this paper by discussing our results and presenting some future work.

2 Related work

Approaches to concept extraction can be categorized by a variety of dimensions including units processed, data sources and knowledge support [20]. The overview of techniques for concept extraction presented in this section focuses on the knowledge support dimension. Accordingly, we differentiate between two main categories of

[1] http://www.nlm.nih.gov/mesh

approaches to concept extraction, namely knowledge-rich and low-bias approaches. Knowledge-rich approaches use knowledge about the structure of the data sources to process. Especially, text-based approaches include knowledge such as phrase structure, lemmas and part-of-speech to extract nouns or noun phrases as units to process [3]. The category of knowledge-rich approaches also includes supervised machine learning techniques and clustering techniques based on knowledge-rich features [11]. Knowledge-rich approaches are subject to limitations regarding their portability to other languages and domains because of the background knowledge they necessitate. Low-bias (also called knowledge-lean [20]) approaches try to remedy these problems by not using a-priori knowledge on the language to process. Rather, they make use of statistical features to extract the features of the terms which compose a concept. Clustering techniques based on low-bias features are the main constituent of this category of approaches.

An early work on low-bias concept extraction considered the use of collocation for measuring the degree of association of words [4]. A similar approach based on head modifiers and modifiers was implemented in [15]. For each term, the number of occurrences as head modifier/modifier of other terms is computed. The resulting vectorial descriptions are compared using the cosine metric. In [17], word vectors are used to describe terms in a corpus. The word vector to each term consist of all its close neighbors, i.e., of all the words which appear in the same sentence or within a larger context (e.g., a document [12]). Since the vectors generated are high-dimensional, salient features are extracted by using the Latent Semantic Analysis (LSA). Then, the cosine metric is applied to the transformed vectors to measure the correlation between the term descriptions. In [16], collocations are use to derive a concept hierarchy from a set of documents. They define a subsumption relation by stating that a term t subsumes a term t', when t appear in every document in which t' appears. Using this subsumption relation, a term hierarchy is computed automatically. A technique that generates concept hierarchies out of document hierarchies is proposed in [8]. The first step of this technique consists of selecting documents from the same domain. Then, a hierarchy of document clusters is generated by using the SOTA-Algorithm [5]. A keyword matching a Wordnet-concept is then assigned bottom-up to each cluster of the hierarchy in two steps: first, a concept representing the typical content of the documents of each leaf node is assigned to the node. In the second step, the labels of the interior nodes are assigned by using hypernyms of their children.

In all the approaches to low-bias concept extraction presented above, the terminology used for extracting concepts is commonly detected using either domain-specific knowledge such as reference vocabularies or language-specific techniques such as deep parsing. In this paper, we present a low-bias approach to concept extraction that makes no use of such a-priori knowledge.

3 An approach to low-bias concept extraction

Our approach is subdivided into two main steps. First, we extract the domain-specific terminology using no a-priori knowledge. Subsequently, we cluster to this terminology to domain-specific concepts.

3.1 Terminology extraction

The extraction of domain-specific terminology is carried out by using a combination of the SRE metric and the SIGNUM algorithm. We use the SRE metric [9] to extract domain-specific multi-word units (MWUs). This metric can efficiently detect domain-specific MWUs by using a combination of the relative expectation of co-occurrences and their distribution over the corpus. The general formula of SRE is given by

$$SRE(w) = \frac{nf(w)p(w)e^{-\frac{(d(w)-\mu)^2}{2\sigma^2}}}{\sqrt{2\pi\sigma^2}\sum_{i=1}^{n} f(c_1 \dots c_i * c_{i+2} \dots c_n)}, \quad (1)$$

where

- $d(w)$ is the number of documents in which w occurs,
- μ and σ^2 are the mean and the variance of the distribution of n-grams in documents respectively,
- $p(w)$ is the probability of occurrence of w in the whole corpus,
- $f(w)$ is the frequency of occurrence of w in the whole corpus and
- $c_1 ... c_i * c_{i+2} ... c_n$ are patterns such that $ham(w, c_1 ... c_i * c_{i+2} ... c_n) = 1$.

The results of SRE can be interpreted as a weighted graph. On this graph, we use SIGNUM [10], a local graph clustering algorithm for terminology extraction. The basic idea behind SIGNUM originates from the spreading activation principle, which has been used in several areas such as neural networks and information retrieval [2]: the simultaneous propagation of information across edges. In the case of SIGNUM, this information consists of the classification of the predecessors of each node in one of the two classes dubbed $+$ and $-$. Each propagation step consists of simultaneously assigning the predominant class of its predecessors to each node. The processing of a graph using SIGNUM thus consists of three phases: the *initialization phase*, during which each node is assigned an initial class; the *propagation phase*, during which the classes are propagated along the edges until a termination condition is satisfied, leading to the *termination phase*. The resulting categorization is then given out.

3.2 Concept extraction

For the extraction of concepts, we represent each of the domain-specific terms included in the terminology extracted priorly by its most significant co-occurrences [7] and compare these representations using the cosine metric. The resulting similarity values are used to compute a term similarity graph, which is used as input for the graph clustering algorithm BorderFlow.

4 BorderFlow

BorderFlow is a general-purpose graph clustering algorithm. It uses solely local information for clustering and achieves a soft clustering of the input graph. The definition of cluster underlying BorderFlow was proposed by [6]. They state that a cluster is a collection of nodes that have more links between them than links to the outside. When considering a graph as the description of a flow system, Flake et al.'s definition of a cluster implies that a cluster X can be understood as a set of nodes such that the flow within X is maximal while the flow from X to the outside is minimal. The idea behind BorderFlow is to maximize the flow from the border of each cluster to its inner nodes (i.e., the nodes within the cluster) while minimizing the flow from the cluster to the nodes outside of the cluster. In the following, we will specify BorderFlow for weighted directed graphs, as they encompass all other forms of non-complex graphs.

4.1 Formal specification

Let G = (V, E, ω) be a weighted directed graph with a set of vertices V, a set of edges E and a weighing function ω, which assigns a positive weight to each edge $e \in E$. In the following, we will assume that non-existing edges are edges e such that $\omega(e) = 0$. Before we describe BorderFlow, we need to define functions on sets of nodes. Let $X \subseteq V$ be a set of nodes. We define the set $i(X)$ of inner nodes of X as:

$$i(X) = \{x \in X | \forall y \in V : \omega(xy) > 0 \rightarrow y \in X\}. \quad (2)$$

The set $b(X)$ of border nodes of X is then

$$b(X) = \{x \in X | \exists y \in V \backslash X : \omega(xy) > 0\}. \quad (3)$$

The set $n(X)$ of direct neighbors of X is defined as

$$n(X) = \{y \in V \backslash X \, | \exists x \in X : \omega(xy) > 0\}. \quad (4)$$

In the example of a cluster depicted in Figure 1, $X = \{3, 4, 5, 6\}$, the set of border nodes of X is $\{3, 5\}$, $\{6, 4\}$ its set of inner nodes and $\{1, 2\}$ its set of direct neighbors.

Let Ω be the function that assigns the total weight of the edges from a subset of V to a subset of V (i.e., the flow between the first and the second subset). Formally:

$$\begin{array}{c} \Omega : 2^V \times 2^V \rightarrow \mathbb{R} \\ \Omega(X, Y) = \sum_{x \in X, y \in Y} \omega(xy). \end{array} \quad (5)$$

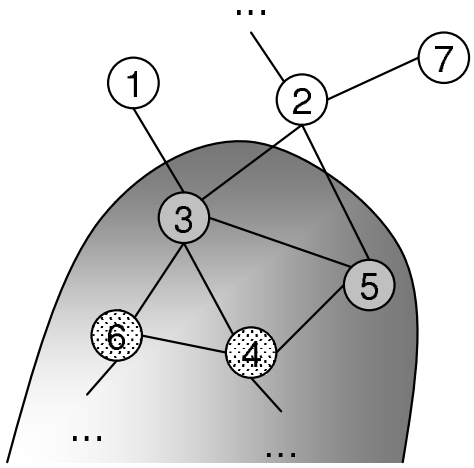

Figure 1: An exemplary cluster. *The nodes with relief are inner nodes, the grey nodes are border nodes and the white are outer nodes. The graph is undirected.*

We define the border flow ratio $F(X)$ of $X \subseteq V$ as follows:

$$F(X) = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), V \backslash X\big)} = \frac{\Omega\big(b(X), X\big)}{\Omega\big(b(X), n(X)\big)}. \quad (6)$$

Based on the definition of a cluster by [6], we define a cluster X as a node-maximal subset of V that maximizes the ratio $F(X)$[2], i.e.:

$$\forall X' \subseteq V, \, \forall v \notin X : X' = X + v \rightarrow F(X') < F(X). \quad (7)$$

The idea behind BorderFlow is to select elements from the border $n(X)$ of a cluster X iteratively and insert them in X until the border flow ratio $F(X)$ is maximized, i.e., until Equation (7) is satisfied. The selection of the nodes to insert in each iteration is carried out in two steps. In a first step, the set $C(X)$ of candidates $u \in V \backslash X$ which maximize $F(X + u)$ is computed is as follows:

$$C(X) := \underset{u \in n(X)}{\arg\max} \; F(X + u). \quad (8)$$

By carrying out this first selection step, we ensure that each candidate node u which produces a maximal flow to the inside of the cluster X and a minimal flow to the outside of X is selected. The flow from a node $u \in C(X)$ can be divided into three distinct flows:

- the flow $\Omega(u, X)$ to the inside of the cluster,
- the flow $\Omega(u, n(X))$ to the neighbors of the cluster and
- the flow $\Omega(u, V \backslash (X \cup n(X)))$ to the rest of the graph.

[2]For the sake of brevity, we shall utilize the notation $X + c$ to denote the addition of a single element c to a set X. Furthermore singletons will be denoted by the element they contain, i.e., $\{v\} \equiv v$.

Prospective cluster members are elements of $n(X)$. To ensure that the inner flow within the cluster is maximized in the future, a second selection step is necessary. During this second selection step, BorderFlow picks the candidates $u \in C(X)$ which maximize the flow $\Omega(u, n(X))$. The final set of candidates $C_f(X)$ is then

$$C_f(X) := \arg\max_{u \in C(X)} \Omega(u, n(X)). \quad (9)$$

All elements of $C_f(X)$ are then inserted in X if the condition

$$F(X \cup C_f(X)) \geq F(X) \quad (10)$$

is satisfied.

4.2 Heuristics

One drawback of the method proposed above is that it demands the simulation of the inclusion of each node in $n(X)$ in the cluster X before choosing the best ones. Such an implementation can be time-consuming as nodes in terminology graphs can have a high number of neighbors. The need is for a computationally less expensive criterion for selecting a nearly optimal node to optimize $F(X)$. Let us assume that X is large enough. This assumption implies that the flow from the cluster boundary to the rest of the graph is altered insignificantly when adding a node to the cluster. Under this condition, the following two approximations hold:

$$\Omega(b(X), n(X)) \approx \Omega(b(X + v), n(X + v)), \quad (11)$$

$$\Omega(b(X), v) - \Omega(d(X, v), X + v) \approx \Omega(b(X), v). \quad (12)$$

Consequently, the following approximation holds:

$$\Delta F(X, v) \approx \frac{\Omega(b(X), v)}{\Omega(b(X + v), n(X + v))}. \quad (13)$$

Under this assumption, one can show that the nodes that maximize $F(X)$ maximize the following:

$$f(X, v) = \frac{\Omega(b(X), v)}{\Omega(v, V \backslash X)} \text{ for symmetrical graphs.} \quad (14)$$

Now, BorderFlow can be implemented in a two-step greedy fashion by ordering all nodes $v \in n(X)$ according to $1/f(X, v)$ (to avoid dividing by 0) and choosing the node v that minimizes $1/f(X, v)$. Using this heuristic, BorderFlow is easy to implement and fast to run.

5 Experiments and results

We evaluated our approach to concept extraction on two data sets of different cleanness and size. In the quantative evaluation, we compared the clustering generated by BorderFlow with that computed using kNN, which is the local algorithm commonly used for clustering tasks. The goal of the qualitative evaluation was to compute the quality of the clusters extracted by using BorderFlow by comparing them with the controlled MESH vocabulary.

5.1 Experimental setup

The data sets underlying the results presented in this chapter are the TREC corpus for filtering [13] and a subset of the articles published by BioMed Central (BMC[3]). Henceforth, we will call the second corpus *BMC*. The TREC corpus is a test collection composed of 233,445 abstracts of publications from the bio-medical domain. It contained 38,790,593 running word forms. The *BMC corpus* consists of full text publications extracted from the BMC Open Access library. The original documents were in XML. We extracted the text entries from the XML data using a SAX[4] Parser. Therefore, it contained a large amount of impurities that were not captured by the XML-parser. The main idea behind the use of this corpus was to test our method on real life data. The 13,943 full text documents contained 70,464,269 running word forms.

The most significant co-occurrences of the terms were computed in two steps. In a first step, we extracted function words by retrieving the f terms with the lowest information content according to Shannon's law [18]. Function words were not considered as being significant co-occurrences. Then, the s best scoring co-occurrences of each term that were not function words were extracted and stored as binary feature vectors.

5.2 Quantitative evaluation

In this section of the evaluation, we compared the average silhouettes [14] of the clusters computed by BorderFlow with those computed by kNN on the same graphs. The silhouette $\sigma(X)$ of a cluster X is given by:

$$\sigma(X) = \frac{1}{|X|} \sum_{v \in X} \frac{a(v, X) - b(v, V \backslash X)}{max\{a(v, X), b(v, V \backslash X)\}}, \quad (15)$$

where

$$a(v, X) = \frac{\sum_{v' \in n(v) \cap X} \omega(v, v')}{|n(v) \cap X|} \quad (16)$$

and

$$b(v, V \backslash X) = \max_{v' \in V \backslash X} \omega(v, v'). \quad (17)$$

To ensure that all clusters had the same maximal size k, we use the following greedy approach for each seed: first, we initiated the cluster X with the seed. Then, we sorted all $v \in n(X)$ according to their flow to the inside of the cluster $\Omega(v, X)$ in the descending order. Thereafter, we sequentially added all v until the size of the cluster reached k. If it did not reached k after adding all neighbors, the procedure was iterated with $X = X \cup n(X)$ until the size k was reached or no more neighbors were found.

One of the drawbacks of kNN lies in the need for specifying the right value for k. In our experiments, we used the average size of the clusters computed using BorderFlow as value for k. This value was 7 when clustering the TREC

[3] http://www.biomedcentral.com

[4] SAX stands for Simple Application Programming Interface for XML.

data. On the BMC corpus, the experiments with $f = 100$ led to $k = 7$, whilst the experiments with $f = 250$ led to $k = 9$. We used exactly the same set of seeds for both algorithms.

The results of the evaluation are shown in Table 1. On both data sets, BorderFlow significantly outperformed kNN in all settings. On the TREC corpus, both algorithms generated clusters with high silhouette values. BorderFlow outperformed kNN by 0.23 in the best case ($f = 100$, $s = 100$). The greatest difference between the standard deviations, 0.11, was observed when $f = 100$ and $s = 200$. On average, BorderFlow outperformed kNN by 0.17 with respect to the mean silhouette value and by 0.08 with respect to the standard deviation. In the worst case, kNN generated 73 erroneous clusters, while BorderFlow generated 10. The distribution of the silhouette values across the clusters on the TREC corpus for $f = 100$ and $s = 100$ are shown in Figure 2(a) for BorderFlow and Figure 2(b) for kNN.

The superiority of BorderFlow over kNN was better demonstrated on the noisy BMC corpus. Both algorithms generate a clustering with lower silhouette values than on TREC. In the best case, BorderFlow outperformed kNN by 0.57 with respect to the mean silhouette value ($f = 250$, $s = 200$ and $s = 400$). The greatest difference between the standard deviations, 0.18, was observed when $f = 250$ and $s = 400$. In average, BorderFlow outperformed kNN by 0.5 with respect to the mean silhouette value and by 0.16 with respect to the standard deviation. Whilst BorderFlow was able to compute a correct clustering of the data set, generating maximally 1 erroneous cluster, using kNN led to large sets of up to 583 erroneous clusters ($f = 100$, $s = 400$). Figures 2(c) and 2(d) show the distribution of the silhouette values across the clusters on the BMC corpus for $f = 100$ and $s = 100$.

5.3 Qualitative evaluation

The goal of the qualitative evaluation was to determine the quality of the content of our clusters. We focused on elucidating whether the elements of the clusters were labels of semantically related categories. To achieve this goal, we compared the content of the clusters computed by BorderFlow with the MESH taxonomy [1]. It possesses manually designed levels of granularity. Therefore, it allows to evaluate cluster purity at different levels. The purity $\varphi(X)$ of a cluster X was computed as follows:

$$\varphi(X) = \max_{C} \left(\frac{|X \cap M|}{|X \cap C^*|} \right), \tag{18}$$

where M is the set of all mesh category labels, C is a MESH category and C^* is the set of labels of C and all its sub-categories. For our evaluation, we considered only clusters that contained at least one term that could be found in MESH.

The results of the qualitative evaluation are shown in Table 2. The best cluster purity, 89.23%, was obtained when clustering the vocabulary extracted from the TREC data with $f = 250$ and $s = 100$. In average, we obtained a lower cluster purity when clustering the BMC data. The best cluster purity using BMC was 78.88% ($f = 100$, $s = 200$). On both data sets, the difference in cluster quality at the different levels was low, showing that BorderFlow was able to detect fine-grained cluster with respect to the MESH taxonomy. Example of clusters computed with $f = 250$ and $s = 400$ using the TREC corpus are shown in Table 3.

6 Discussion

From a quantitative point of view, the average silhouette values μ on TREC were higher with lower standard deviations σ. The difference in silhouette can be conceivably explained by the higher amount of noise contained in the BMC corpus. On the TREC corpus, a higher size of the feature vectors led to a higher value μ of the average silhouette of the clusters. The same relation could be observed between the number f of function words omitted and the value of μ. The standard deviation σ was inversely proportional to the size of the feature vectors and the number of function words. The number of erroneous clusters (i.e., clusters with average silhouette value less than 0) was inversely proportional to the size of the feature vectors. This can be explained by the higher amount of information available, which led to a better approximation of the semantic similarity of the terms and, thus, to less clustering mistakes. In the worst case (f=100, s=100), 99.85% of the clusters had positive silhouettes. From a qualitative point of view, BorderFlow computed clusters with a high purity based on low-level features extracted on a terminology extracted using low-bias techniques. As expected, the average cluster purity was higher for clusters computed using the TREC data set. The results of the qualitative evaluation support the basic assumption underlying this work, i.e., it is indeed possible to extract high-quality concepts from text automatically without a-priori knowledge.

Acknowledgement

This work was supported by a grant of the German Ministry for Education and Research.

References

[1] S. Ananiadou and J. Mcnaught. *Text Mining for Biology and Biomedecine*. Norwood, MA, USA, 2005.

[2] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[3] C. Biemann. Ontology learning from text: A survey of methods. *LDV Forum*, 20(2):75–93, 2005.

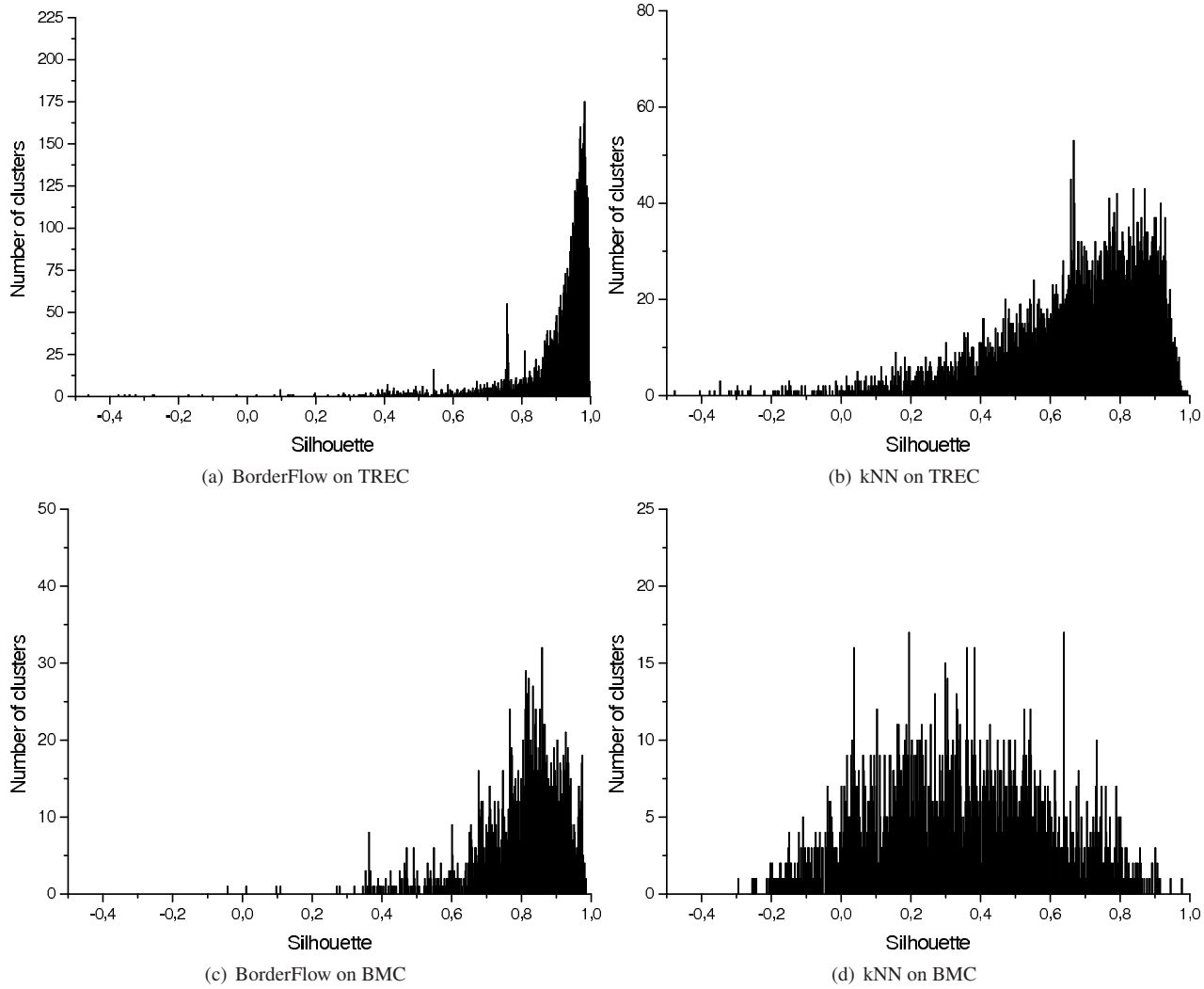

Figure 2: Distribution of the average silhouette values obtained by using BorderFlow and kNN on the TREC and BMC data set with f=100 and s=100

		$\mu \pm \sigma$				Erroneous clusters			
f	s	TREC		BMC		TREC		BMC	
		kNN	BF	kNN	BF	kNN	BF	kNN	BF
100	100	0.68±0.22	**0.91**±0.13	0.37±0.28	**0.83**±0.13	73	**10**	214	**1**
100	200	0.69±0.22	**0.91**±0.11	0.38±0.27	**0.82**±0.12	68	**1**	184	**1**
100	400	0.70±0.20	**0.92**±0.11	0.41±0.26	**0.83**±0.12	49	**1**	142	**1**
250	100	0.81±0.17	**0.93**±0.09	0.23±0.31	**0.80**±0.14	10	**2**	553	**0**
250	200	0.84±0.13	**0.94**±0.08	0.23±0.31	**0.80**±0.14	5	**2**	575	**0**
250	400	0.84±0.12	**0.94**±0.08	0.24±0.32	**0.80**±0.14	2	**1**	583	**0**

Table 1: Comparison of the distribution of the silhouette index over clusters extracted from the TREC and BMC corpora. BF stands for BorderFlow. μ the mean of silhouette values over the clusters and σ the standard deviation of the distribution of silhouette values. Erroneous clusters are cluster with negative silhouette silhouettes. Bold fonts mark the best results in each experimental setting.

[4] Kenneth W. Church and Patrick Hanks. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th. Annual Meeting of the Association for Computational Linguistics*, pages 76–83, Vancouver, B.C., 1989. Association for Computational Linguistics.

	f=100	f=100	f=100	f=250	f=250	f=250
Level	s=100	s=200	s=400	s=100	s=200	s=400
1	86.81	81.84	81.45	89.23	87.62	87.13
2	85.61	79.88	79.66	87.67	85.82	86.83
3	83.70	78.55	78.29	86.72	84.81	84.63
1	78.58	78.88	78.40	72.44	73.85	73.03
2	76.79	77.28	76.54	71.91	73.27	72.39
3	75.46	76.13	74.74	69.84	71.58	70.41

Table 2: Cluster purity obtained using BorderFlow on TREC and BMC data. *The upper section of the table displays the results obtained using the TREC corpus. The lower section of the table displays the same results on the BMC corpus. All results are in %.*

Cluster members	**Seeds**	**Hypernym**
b_fragilis, c_albicans, candida_albicans, l_pneumophila	c_albicans	Etiologic agents
acyclovir, methotrexate_mtx, mtx, methotrexate	methotrexate	Drugs
embryo, embryos, mouse_embryos, oocytes	embryo, embryos, mouse _embryos, oocytes	Egg cells
leukocytes, macrophages, neutrophils, platelets, pmns	platelets	Blood cells
flap, flaps, free_flap, muscle_flap, musculocutaneous_flap	flap, free_flap	Flaps
leukocyte, monocyte, neutrophil, polymorphonuclear_leukocyte	polymorphonuclear_leukocyte	White blood cells

Table 3: Examples of clusters extracted from the TREC corpus. The relation between the elements of the clusters is displayed in the rightmost column. Cluster members in italics are erroneous.

[5] J. Dopazo and J. M. Carazo. Phylogenic reconstruction using a growing neural network that adopts the topology of a phylogenic tree. *Molecular Evolution*, 44:226–233, 1997.

[6] G. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Proceedings of the 6th ACM SIGKDD*, pages 150–160, Boston, MA, 2000.

[7] G. Heyer, M. Läuter, U. Quasthoff, T. Wittig, and C. Wolff. Learning relations using collocations. In *Workshop on Ontology Learning*, volume 38 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2001.

[8] Latifur Khan and Feng Luo. Ontology construction for information selection. In *Proceedings of the ICTAI'02*, pages 122–127, Washington DC, USA, 2002. IEEE Computer Society.

[9] A.-C. Ngonga Ngomo. Knowledge-free discovery of multi-word units. In *Proceedings of the 23rd Annual ACM Symposium on Applied Computing*, pages 1561–1565. ACM Press, 2008.

[10] A.-C. Ngonga Ngomo. SIGNUM: A graph algorithm for terminology extraction. In *Proceedings of CICLing' 2008*, pages 85–95. Springer, 2008.

[11] B. Omelayenko. Learning of ontologies for the web: the analysis of existent approaches. In *Proceedings of the International Workshop on Web Dynamics*, pages 268–275, 2001.

[12] Yonggang Qiu and Hans-Peter Frei. Concept-based query expansion. In *Proceedings of SIGIR'93*, pages 160–169, Pittsburgh, US, 1993.

[13] Stephen E. Robertson and David Hull. The TREC 2001 filtering track report. In *Proceedings of the Text REtrieval Conference*, 2001.

[14] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(1):53–65, 1987.

[15] G. Ruge. Experiments on linguistically-based term associations. *Information Processing and Management*, 28(3):317–332, 1992.

[16] M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *Proceedings of SIGIR '99*, pages 206–213, New York, NY, USA, 1999. ACM.

[17] H. Schütze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.

[18] C. E. Shannon. A mathematic theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.

[19] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 1 edition, 2005.

[20] L. Zhou. Ontology learning: state of the art and open issues. *Information Technology and Management*, 8(3):241–252, 2007.