Evaluation of Technology Term Recognition with Random Indexing

Behrang Q. Zadeh and Siegfried Handschuh
Knowledge Discovery Unit
Insight Centre for Data Analytics
National University of Ireland, Galway
behrang.qasemizadeh@insight-centre.org, siegfried.handschuh@insight-centre.org

Abstract

In this paper, we propose a method that combines the principles of automatic term recognition and the distributional hypothesis to identify technology terms from a corpus of scientific publications. We employ the random indexing technique to model terms' surrounding words, which we call the context window, in a vector space at reduced dimension. The constructed vector space and a set of reference vectors, which represents manually annotated technology terms, in a k-nearest-neighbour voting classification scheme are used for term classification. In this paper, we examine a number of parameters that influence the obtained results. First, we inspect several context configurations, i.e. the effect of the context window size, the direction in which co-occurrence counts are collected, and information about the order of words within the context windows. Second, in the k-nearest-neighbour voting scheme, we study the role that neighbourhood size selection plays, i.e. the value of k. The obtained results are similar to word space models. The performed experiments suggest the best performing context are small (i.e. not wider than 3 words), are extended in both directions and encode the word order information. Moreover, the accomplished experiments suggest that the obtained results, to a great extent, are independent of the value of k.

Keywords: Automatic Term Recognition and Classification, Distributional Semantics, Random Indexing

1. Introduction

Successful identification of terms that correspond to technology concepts is a primarily step towards gaining knowledge about new technological developments and innovations that are continuously reported in the scientific literature. In this paper, we propose technology term recognition (TTR) to identify these terms. TTR can be viewed as a kind of automatic term recognition (ATR). ATR is the task of identification of domain-specific terms. The main goal of ATR is to determine whether a word or phrase is a term that characterizes a target domain (Frantzi et al., 1998). TTR, however, targets a subset of terms that characterizes technologies in a target domain. If T_{tech} is the set of all the technology terms in a domain extracted by a TTR system, and T_{term} is the set of all the terms in a domain that are recognized by an ATR system, then we expect that $T_{tech} \subset T_{term}$ (Figure 1). For instance, in computational linguistics literature both *language resources* and *natural language processing* are valid terms; however, TTR must only recognize the latter as a valid technology term.

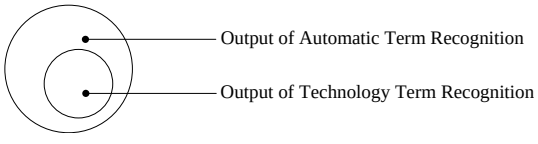

Figure 1: Comparison of ATR and TTR outputs.

Defining technology – and subsequently finding its corresponding terms – is a complex task and has been the subject of study in a number of disciplines such as philosophy of science (Mitcham and Briggle, 2012). In this paper, we avoid searching for an analytical answer to the question: "what technology is". Instead of relying on a formal definition for technology, we exploit the context of terms in order to identify technology terms among them. We trust that technology terms tend to appear in similar linguistic contexts and by extending Harris's (1954) Distributional Hypothesis, we claim that the context of (previously) known technology terms can be modelled and used in order to identify new unknown technology terms.

The proposed method for TTR is realized as a term classification task on top of a generic ATR framework. The algorithms for ATR are usually in the form of a two-step procedure: (a) candidate term extraction followed by (b) term scoring and ranking (Nakagawa, 2001). The candidate term extraction deals with the term formation and the extraction of term candidates. The subsequent scoring procedure, however, can be seen as a semantic weighting mechanism. As Figure 2 suggests, in order to indicate how likely it is that a candidate term is a term we would like to extract, the scoring procedure assigns a weight to each of the extracted candidate terms, usually as a combination of scores known as termhood and unithood (Kageura and Umino, 1996).

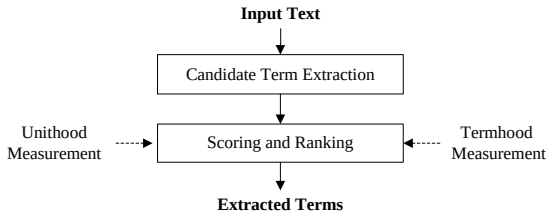

Figure 2: General architecture of an ATR system.

Following the extraction of candidate terms and allocating scores, TTR employs the contextual similarity of terms in order to measure their relevance to technology concepts. We assume that the association of a term to a technology concept is a kind of *paradigmatic relation* that can be characterized using the *syntagmatic relations* of the term and its surrounding words. The major research question to be

answered here, therefore, is the definition of terms' contexts and the examination of their distributional properties. We suggest that the co-occurrence of a term and words in a window of text around the term's appearances in the corpus defines the contexts for the described paradigmatic relation. The major parameters to be examined for such a context are thus the size of the co-occurrence region, the position of the term in the context window and the direction in which the neighbourhood is extended (Lenci, 2008).

In this paper, the context windows surrounding candidate terms are represented using vector space models (VSMs). A k-nearest-neighbour (k-nn) voting classification framework is then employed to identify the technology terms. We construct the VSMs at reduced dimension using the *random indexing* (RI) technique (Sahlgren, 2005). Context windows are configured using various sizes and directions, as well as the words' order information. Each configuration is represented by a VSM. In order to suggest the best context window configuration for the TTR task, the performance of each of the constructed VSMs in the k-nn classification framework is examined and reported.

In the remaining sections, we briefly introduce the RI technique in Section 2. The term recognition and classification framework is described in Section 3. Section 4 explains the evaluation framework and metrics and reports the obtained results. We conclude this paper in Section 5.

2. Random Indexing

For a corpus of a relatively small size, the context of terms may be defined and examined efficiently using a conventional method of vector space construction, presumably followed by a method of dimensionality reduction. In such methods, a new context is appended to a VSM by increasing the dimension of the VSM, i.e. by adding a new element to the standard basis of the VSM. As the corpus grows, however, due to the power-law distribution of contexts over terms, the number of contexts that represent terms bursts. The result is a VSM of a very high dimension – on the orders of hundreds of thousands or more. The high dimension of the VSM precipitates low computational performance in the subsequent processes such as dimension reduction and similarity measurements. RI alleviates this problem by combining the construction of the vector space and the dimension reduction process.

The RI method, introduced by Kanerva et al. (2000), constructs a VSM at reduced dimension without requiring prior construction of the VSM at its original high dimension. Using the Johnson and Lindenstrauss (1984) lemma and basic properties of matrix arithmetic, it can be verified that RI is a dimension reduction technique that employs a sparse random projection. The RI method maps p vectors in an n-dimensional VSM to an $m \geq m_0 = \log(p/\epsilon^2)$ dimensional subspace, for $n \gg m$, while it preserves pairwise *Euclidean* distances between vectors by the small error factor of $1 \pm \epsilon$ and a high probability.

Sahlgren (2005) delineates the RI technique as a two-step procedure: (a) the creation of *index vectors* and (b) the construction of *context vectors*.

In the first step, each context is *uniquely* assigned to an *index vector*, i.e. a randomly generated high-dimensional vector where most of the elements of the vectors are set to 0 and only a few to 1 and -1. Following the proofs given by Li et al. (2006), we suggest random index vectors with i.i.d. entries r_i such that

$$r_{ij} = \sqrt{s} \begin{cases} -1 & \textit{with probability } \frac{1}{2s} \\ 0 & \textit{with probability } 1 - \frac{1}{s} \\ 1 & \textit{with probability } \frac{1}{2s} \end{cases} \quad (1)$$

for $s = \sqrt{n}$, where n is the number of employed contexts for the construction of the VSM.

In the second step, each target entity, which is a term candidate in our experiment, is assigned to a vector, called a *context vector*. The context vector has the same dimension as index vectors have, and all of its elements, initially, are set to 0. For each co-occurrence of an entity and a context, e.g. through a sequential scan of the corpus, the context vector $\mathbf{v}_e$ that represents the entity is accumulated by the index vector $\mathbf{r}_i$ that represents the context, i.e. $\mathbf{v}_e = \mathbf{v}_e + \mathbf{r}_i$. The procedure results in a VSM that represents entities using the employed contexts, however, at reduced dimension.

The RI method offers a number of benefits compared to a classic vector space construction method that is followed by a *data-sensitive* dimension reduction technique such as truncated singular value decomposition (SVD). At least, the RI eliminates the need for the dimension reduction that requires a process of a computational complexity not better than $\mathrm{O}(nm)$.

As suggested earlier, the RI method is justified by mathematical principles. In addition, several empirical experiments prove its viability. In an earlier reported experiment, in a document similarity measurement application, Bingham and Mannila (2001) show that dimension reduction using a sparse random projection, which is comparable to the RI technique, provides similar results to the truncated SVD. In addition, a growing amount of research in diverse application domains has successfully employed the RI method for the construction of VSMs at reduced dimension (see e.g. Baroni et al., 2007; Sahlgren and Karlgren, 2009; Cohen et al., 2010; Jurgens and Stevens, 2010; Yannakoudakis and Briscoe, 2012).

3. Term Recognition and Classification

We design our method for technology term recognition on top of a generic automatic term recognition (ATR) process. In the first step of the process, we employ linguistic filters in the form of part-of-speech (PoS) tag sequence patterns to extract candidate terms. The employed method is similar to the one proposed by Justeson and Katz (1995); however, we restrict the length of candidate terms to up to six constituent words and allow for 44 different PoS patterns. In addition, in order to lessen the effect of erroneous automatic PoS tagging, similar to the proposed method by Ittoo et al. (2010), we formulate the PoS patterns by observing the actual output of the employed PoS tagger and allowing the patterns that contain erroneous PoS tags.

Following the extraction of candidate terms, we employ the c-value algorithm to assign scores to the terms. For each candidate term t, the c-value score of t, $c\text{-}value(t)$, is calculated using four criteria (Frantzi et al., 1998): the frequency

of t in the corpus; the frequency of t when it appears nested in other terms longer than t; the number of those longer terms shown by T_t; and the number of the constituent words of t shown by $|t|$. The c-value score is given by

$$c\text{-}value(t) = \begin{cases} \log_2 |t| f(t) & \text{if } t \notin \text{nested} \\ \log_2 |t| \left(f(t) - \frac{1}{|T_t|} \sum_{b \in T_t} f(b)\right) & \text{otherwise} \end{cases},$$

where T_t denotes the set of all the terms that contain t and are longer than t, and $f(s)$ denotes the frequency of an arbitary term s in the corpus.

Following to ATR, we employ a k-nn voting scheme to approximate the association of terms to technology concepts. We employ the random indexing technique to construct context vectors for the candidate terms. We calculate the cosine similarity of the context vector v_t of a target term t to a set of reference vectors R_s. Valid technology terms in the R_s – terms that correspond to a technological concept – are manually marked by an expert prior to the similarity calculations. After the similarity calculation, vectors in the R_s are sorted in descending order by their similarity to v_t. We count the number of valid technology terms, which is shown by $|T_{tech}|$, in the top k terms of this sorted list. The weight $w(t) = {}^{|T_{tech}|}/_{k}$ is then considered as a measurement of the technology-term class membership for t.

To get the final ranking score of a candidate term in the corpus, we combine its assigned c-value score and the technology-term class membership weight. Terms are first ranked and sorted in descending order by their assigned technology-term class membership weight. Terms with equal class membership weight are then sorted in descending order by their c-value scores. Although other combinations of the c-value score and the classification weight can enhance the ranking, e.g. a linear combination as suggested in Maynard and Ananiadou (2000), the proposed ranking technique provides an immaculate framework to evaluate the role that contexts plays in the classification task.

4. Evaluation Framework

4.1. Benchmark Data and Baseline

We have created a benchmark dataset to evaluate the task of terminology extraction and classification. The dataset is a spin-off from the ACL anthology reference corpus (ACL ARC). The ACL ARC is a frozen canonicalized corpus of scholarly publications for bibliographic research in computational linguistics (Bird et al., 2008). In order to prepare the benchmark data, the ACL ARC corpus is first converted to raw text files.[1] We further employ the SectLabel module of the ParsCit tool to identify logical sections in the raw text files and remove irrelevant sections such as bibliography and acknowledgements from the text that is being analyzed (Luong et al., 2010).[2] The sectioning process is followed by the segmentation of text using the OpenNLP sentence splitter[3] and the Stanford tokenizer and part-of-speech tagger (Toutanova et al., 2003).[4] Afterwards, the candidate terms are extracted from the processed corpus. The process is finalized by assigning unique identifiers to the extracted linguistic units such as words, sentences, and candidate terms. The statistics of the resulting data are given in Table 1.

Type	Token	Sentence	Paragraph	Section	Can-term
704,085	36,729,513	1,564,430	510,366	92,935	1,339,773

Table 1: Summary statistics of the benchmark dataset: each column shows the number of linguistic units in the benchmark dataset extracted from the 10,922 publications in the ACL ARC corpus. The last column, which is marked as Can-term, shows the number of extracted candidate terms.

The described procedure of term recognition and classification in Section 3 is applied to the extracted candidate terms. In each experiment, candidate terms are sorted in descending order by their assigned weights. The top 1000 terms are manually annotated as *valid* and *invalid* terms. Valid terms are additionally categorized as *technology* and *non-technology terms* by the authors of the paper. The proportion of valid technology terms to the number of all candidate terms in the list of the top n terms ($n = 100, 250, 1000$) as well as the proportion of valid terms in the list are reported for the comparison of the evaluated contexts. As for the baseline, the above numbers are also reported for the list of candidate terms sorted in descending order by their c-value score.

4.2. Reference Vector Space Formation

The set of the reference vectors R_s is made by the manual annotation of a subset of candidate terms. Those terms that terminate or collocate with the lemmas *technology* and *technique* are chosen as reference terms. The selected terms are manually annotated as valid and invalid technology terms. In each evaluation scenario, the relevant context vectors of these terms define the R_s. This procedure results in 3490 terms, including 1596 terms that are annotated as valid technology terms.

4.3. Evaluated Contexts

The context vectors of candidate terms are made using the part-of-speech tagged words in their neighbourhood in the whole corpus – which we will call *context window* from now on.. We evaluate our proposed method for term classification using context windows that are configured with three elements: direction, size and order. The first element distinguishes context windows according to the direction in which they are expanded to collect the co-occurrence counts.

The context window of a term is expanded (a) to the *left*-hand side of the term to count the co-occurrences of the term with its preceding words in each sentence of the corpus, (b) to the *right*-hand side to collect co-occurrences with the succeeding words or (c) *around* the term, i.e. in both left and right directions.

The context windows are also configured by their size, i.e. the extent of terms' neighbourhood for counting the co-occurrences. As restated by Sahlgren (2008), optimum size

[1] The raw text files are extracted using Apache PDFBox (`http://pdfbox.apache.org/`), release version 1.7.1.

[2] Release version 110505.

[3] Release version 1.5.2 (`http://opennlp.apache.org/`).

[4] Release date 9 July 2012.

of context window can only be established through experiments. However, he also indicates that narrower context windows are more suitable to capture a paradigmatic relation such as the term classification task proposed here. For this reason, in our experiments we limit the size of context windows to $n \in \{1, 2, 3, 4, 5\}$. For the context windows that expand around a term, we extend the context region symmetrically in both directions.

Jones and Mewhort (2007) state that the sequential order of words expresses information about lexical classes and grammatical behaviour and, therefore, is important in the development of a comprehensive distributional semantic space. On the other hand, Landauer (2002) argues that 80% of the potential information in language is carried by the word choice regardless of the order in which they appear. Landauer thus concludes that word order can be neglected in order to simplify the construction of VSMs and their subsequent computations. We investigate the impact of word order information on the performance of the suggested task using the permutation technique (Sahlgren et al., 2008).

In the permutation technique, the order of words in a context window is captured by shuffling their index vectors via a permutation function. The permutation function is defined using the location of the context words in the contex window. The main idea is that a permutation of randomly created index vectors also creates new random vectors that can be used to represent context words at specific locations in the context windows. In our implementation, a circular shift function serves as the permutation function. If t is the number of tokens after/before a target term and a context word, then the index vector of context word is shifted t times circularly to the right/left before its addition to the target term's context vector.

4.4. Other Evaluation Parameters

In addition to various configurations of context windows, we investigate the effect of the neighbourhood size selection, i.e. the value of k, on the performance of the proposed k-nn voting classification scheme. In this framework, a small value for k leads to *over-fitting*, while a large neighbourhood estimation may reduce the discriminatory power of the classifier. Therefore, the optimal value of k is usually obtained by an experimental method. Yang (1999) suggests that the performance of k-nn is relatively stable for a large range of k. Accordingly, in this case study, the size of neighbourhood is defined to be $k = \lfloor p|R_s| \rfloor$ where $p \in \{0.001, 0.005, 0.01, 0.10, 0.20\}$ and $|R_s|$ is the number of reference vectors, i.e., as described in Section 4.2, $|R_s| = 3490$. We are interested to see if the choice of k affects our choice for the best performing context window.

We performed our evaluation by setting the dimension of the random indices, thus the VSM, to 1800, which should be high enough to embed 1 339 773 vectors that represent candidate terms. The effective number of context words in our experiment is counted to be less than 634 294, i.e. the number of unique part-of-speech tagged words that appear adjacent to candidate terms. We set the value of s in Equation 1 to 225, which is less than $\sqrt{634\,294} = 796$ and thus a safe choice for the constructed VSMs in our experiments. We employ the cosine measure to calculate similarity between vectors. In the reported results, we *do not* employ a weigh normalization process.

Context		Top 100		Top 250		Top 1000	
Type	Size	T_{Term}	V_{Term}	T_{Term}	V_{Term}	T_{Term}	V_{Term}
Left	1	0.600	0.720	0.632	0.728	0.514	0.608
	2	0.570	0.700	0.532	0.664	0.419	0.540
	3	0.500	0.680	0.512	0.636	0.374	0.486
	4	0.430	0.540	0.404	0.504	0.379	0.492
	5	0.400	0.500	0.416	0.496	0.380	0.486
Right	1	0.830	0.850	0.660	0.772	0.520	0.633
	2	0.650	0.740	0.536	0.620	0.375	0.499
	3	0.610	0.670	0.344	0.460	0.239	0.366
	4	0.530	0.650	0.424	0.572	0.345	0.482
	5	0.460	0.620	0.336	0.472	0.278	0.401
Around	1	0.860	0.920	0.692	0.784	0.524	0.650
	2	0.740	0.810	0.620	0.720	0.472	0.585
	3	0.830	0.860	0.656	0.720	0.491	0.587
	4	0.800	0.850	0.588	0.660	0.448	0.550
	5	0.670	0.730	0.428	0.496	0.399	0.494
$Left_{\prod}$	2	0.650	0.740	0.584	0.692	0.479	0.611
	3	0.570	0.650	0.580	0.704	0.481	0.624
	4	0.560	0.660	0.572	0.652	0.481	0.606
	5	0.610	0.690	0.524	0.616	0.480	0.600
$Right_{\prod}$	2	0.840	0.860	0.712	0.756	0.452	0.562
	3	0.720	0.760	0.592	0.684	0.399	0.557
	4	0.520	0.660	0.512	0.664	0.353	0.507
	5	0.510	0.630	0.420	0.568	0.316	0.474
$Around_{\prod}$	1	0.760	0.890	0.748	0.916	0.618	0.786
	2	0.850	0.940	0.784	0.904	0.638	0.773
	3	0.910	0.950	0.780	0.896	0.630	0.748
	4	0.890	0.940	0.732	0.832	0.615	0.731
	5	0.860	0.920	0.688	0.808	0.591	0.702
c-value		0.300	0.760	0.252	0.732	0.202	0.564

Table 2: The observed results in the performed evaluations for various context windows. $\prod$ denotes context types that capture the word order information using the permutation technique. The best observed results for T_{Term} and V_{Term} are marked respectively by an ellipse and rectangle around them. The last row of the table reports these numbers for the baseline measure, i.e. the c-value score.

4.5. Results

Table 2 shows the results obtained from our experiments. In this table, T_{Term} shows the proportion of the number of valid technology terms to the top n terms from the list of candidates that are sorted in descending order by their assigned weights for $n \in 100, 250, 1000$. Similarly, T_{Term} shows the proportion of valid terms in this list. In this experiment, the weights are obtained when k in the k-nn classification scheme is set to $k = \lfloor {}^{|R_s|}/_{100} \rfloor = 34$.

Figure 3 plots the results obtained for each context window size. As can be seen, the $Around_{\prod}$ context window, which extends around candidate terms and incorporates word order information, consistently shows better performance than other context types. As expected, for the context type $Around_{\prod}$, the best choice of context size is 2 or 3 words around the terms. However, for the other context window types, surprisingly, the context windows that extend only to 1 word around, after or before a term show better performance than context windows of larger size 2 or 3, even when the word order is included in the model. In all the experiments, the observed results are consistently above the baseline with a large margin.

As can be inferred from Figure 3, choosing the best performing context by looking at the list of top n terms is subject to the value of n. We expect that the larger values of n

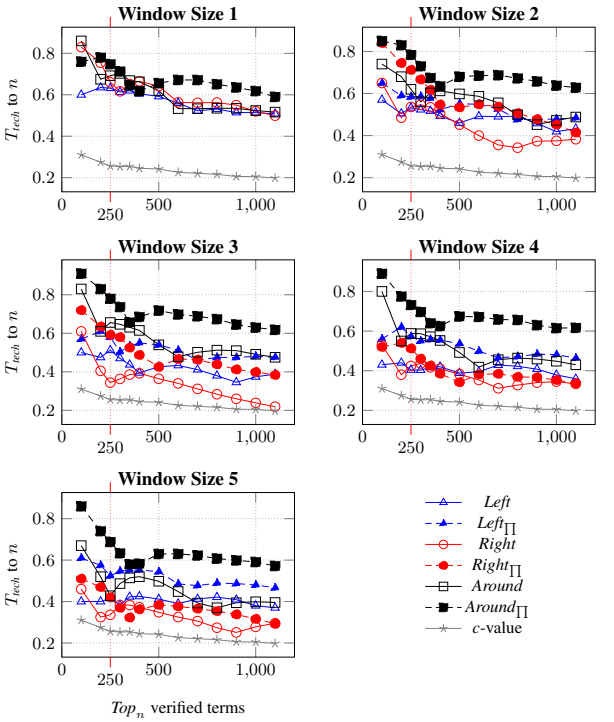

Figure 3: Comparison of the performance of context windows of various size. This visualizes the numbers reported numbers in Table 2.

Context		Value of p in k $= \lfloor p\|R_s\| \rfloor$				
Type	**Size**	**0.001**	**0.005**	**0.01**	**0.10**	**0.20**
Left	**1**	0.580	0.596	0.632	0.412	0.352
	2	0.516	0.560	0.532	0.428	0.216
	3	0.548	0.500	0.512	0.364	0.268
	4	0.556	0.496	0.404	0.356	0.212
	5	0.572	0.456	0.416	0.272	0.212
Right	**1**	0.472	0.588	0.660	0.500	0.184
	2	0.588	0.588	0.536	0.292	0.224
	3	0.584	0.524	0.344	0.236	0.136
	4	0.608	0.500	0.424	0.224	0.164
	5	0.544	0.488	0.336	0.256	0.200
Around	**1**	0.584	0.716	0.692	0.408	0.384
	2	0.612	0.752	0.620	0.372	0.216
	3	0.628	0.700	0.656	0.356	0.232
	4	0.124	0.688	0.588	0.256	0.208
	5	0.168	0.572	0.428	0.240	0.184
$Left_{\prod}$	**2**	0.548	0.552	0.584	0.404	0.320
	3	0.528	0.628	0.580	0.312	0.272
	4	0.540	0.584	0.572	0.272	0.176
	5	0.548	0.568	0.524	0.280	0.160
$Right_{\prod}$	**2**	0.652	0.628	0.712	0.212	0.096
	3	0.592	0.412	0.592	0.196	0.168
	4	0.604	0.428	0.512	0.144	0.160
	5	0.560	0.440	0.420	0.140	0.104
$Around_{\prod}$	**1**	0.600	0.816	0.748	0.652	0.448
	2	0.588	0.820	0.784	0.560	0.340
	3	0.580	0.820	0.780	0.556	0.324
	4	0.592	0.840	0.732	0.416	0.248
	5	0.612	0.804	0.688	0.400	0.204

Table 3: Evaluation of the neighbourhood size selection in the performance of the proposed k-nn classification scheme. The employed values of k are represented as a proportion p of the number of reference vectors $|R_s|$. The best performing context type and size is marked by a rectangle.

give a more realistic measure of the performance than the smaller values of n; however, inspecting large numbers of n terms requires substantial manual annotations, which is cumbersome. In this experiment, we observe that the performance of context windows with respect to each other is nearly constant when $n \geq 250$. As a result, in the next experiment, the performance of the k-nn for different values of k is assessed by looking at the top 250 terms.

Table 3 shows the observed results in the classification task when k is set to various percentages of the size of reference vector size, as explained in Section 4.4. It is important to note that except for very small or large values of k, i.e. when $p \leq 0.001$ or $p \geq 0.20$, the various sizes of the $Around_{\prod}$ context outperforms other context types. Therefore, although the overall performance of the classification task is subject to the value of k, we conclude that the most discriminative context type can be decided independent of the value of k. For this context type, it seems that the larger values of k tend to give better results for the smaller context sizes.

4.6. Replicating the Experiments

The processed corpus and the annotation files can be obtained from the European Language Resources Association (ELRA), catalogue reference ELRA-T0375, the ACL RD-TEC: A Reference Dataset for Terminology Extraction and Classification Research in Computational Linguistics. The dataset comprises 67 641 annotated candidate terms, where 19 223 are valid terms, including 13 010 technology terms.

5. Conclusions

We examined different context types of various sizes in order to find the most discriminative model in a term classification task. We employed the random indexing technique to construct vector space models at reduced dimension. The term classification task is performed using a k-nearest neighbour voting framework. In our experiment, models that are induced from the co-occurrences of terms in context windows that extend to both sides of the terms and encode the order of words outperform other evaluated contexts. In addition, our experiment suggests that the most discriminative context can be identified, to a large extent, independent of the neighbourhood size k in the classification task.

The presented research can be extended in several ways. We did not apply any weighting process prior to the similarity measurements, which can affect the obtained results and performances. The employed cosine similarity can be compared with the Euclidean distance when the vectors are normalized using various weighting techniques. In addition, the binary voting scheme for the classification task can be replaced by the weighted sum of similarities. Alternatively, machine learning techniques other than k-nn, e.g. support vector machines, may be employed to classify terms. Last but not least, the effect of the changes in the set of reference vectors, e.g. its size, on the performance of the classification task can be assessed in future work.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

References

Baroni, M., Lenci, A., and Onnis, L. (2007). ISA meets Lara: An incremental word space model for cognitively plausible simulations of semantic learning. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*. Czech Republic: ACL.

Bingham, E. and Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the Seventh ACM SIGKDD KDD '01*. New York, NY, USA: ACM.

Bird, S., Dale, R., Dorr, B., Gibson, B., Joseph, M., Kan, M. Y., Lee, D., Powley, B., Radev, D., and Tan, Y. F. (2008). The ACL anthology reference corpus: A reference dataset for bibliographic research in computational linguistics. In *LREC'08*. Marrakech, Morocco.

Cohen, T., Schvaneveldt, R., and Widdows, D. (2010). Reflective random indexing and indirect inference: A scalable method for discovery of implicit connections. *Journal of Biomedical Informatics*, 43(2):240 – 256.

Frantzi, K., Ananiadou, S., and Tsujii, J. (1998). The c-value/nc-value method of automatic recognition for multi-word terms. In *Research and Advanced Technology for Digital Libraries*, volume 1513 of *LNCS*. Springer, 585–604.

Harris, Z. S. (1954). Distributional structure. *Word, The Journal of the Inter. Linguistic Association*, 10:146–162.

Ittoo, A., Maruster, L., Wortmann, H., and Bouma, G. (2010). Textractor: A framework for extracting relevant domain concepts from irregular corporate textual datasets. In *BIS*, volume 47 of *LNBIP*. Springer.

Johnson, W. and Lindenstrauss, J. (1984). Extensions of Lipschitz mappings into a Hilbert space. In *Conference in modern analysis and probability (New Haven, Conn., 1982)*, volume 26 of *Contemporary Mathematics*. AMS, pages 189–206.

Jones, M. N. and Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114:1–37.

Jurgens, D. and Stevens, K. (2010). HERMIT: Flexible clustering for the SemEval-2 WSI Task. In *Proceedings of SemEval '10*. Stroudsburg, PA, USA: ACL.

Justeson, J. S. and Katz, S. M. (1995). Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1.1:9–27.

Kageura, K. and Umino, B. (1996). Methods of automatic term recognition: A review. *Terminology*, 3.2 (1996):259–289.

Kanerva, P., Kristoferson, J., and Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*. Erlbaum.

Landauer, T. K. (2002). On the computational basis of learning and cognition: Arguments from LSA. *The Psychology of Learning and Motivation*, 41:43–84.

Lenci, A., 2008. Distributional semantics in linguistic and cognitive research. *From context to meaning: Distributional models of the lexicon in linguistics and cognitive science, special issue of the Italian Journal of Linguistics*. 20/1:1–31.

Li, P., Hastie, T. J., and Church, K. W. (2006). Very sparse random projections. In *Proceedings of the KDD '06*. New York, NY, USA: ACM.

Luong, M.-T., Nguyen, T. D., and Kan, M.-Y. (2010). Logical structure recovery in scholarly articles with rich document features. *IJDLS*, 1(4):1–23.

Maynard, D. and Ananiadou, S. (2000). Identifying terms by their family and friends. In *Proceedings of the 18th Conference on Computational Linguistics - volume 1*, COLING '00. Stroudsburg, PA, USA: ACL.

Mitcham, C. and Briggle, A. (2012). Theorizing technology. *The Good Life in a Technological Age*. Routledge.

Nakagawa, H. (2001). Automatic term recognition based on statistics of compound nouns. *Terminology*, 6.2:195–210.

Sahlgren, M. (2005). An introduction to random indexing. In *Methods and Applications of Semantic Indexing Workshop at the TKE 2005. TermNet News*, 87: 1–9.

Sahlgren, M. (2008). The distributional hypothesis. *Italian Journal of Linguistics*, 20:33–54.

Sahlgren, M., Holst, A., and Kanerva, P. (2008). Permutations as a means to encode order in word space. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*. Austin, TX: Cognitive Science Society, pages 1300–1305.

Sahlgren, M. and Karlgren, J. (2009). Terminology mining in social media. In *Proceedings of the CIKM '09*. New York, NY, USA: ACM.

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the NAACL '03 - Volume 1*. Stroudsburg, PA, USA: ACL.

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90.

Yannakoudakis, H. and Briscoe, T. (2012). Modeling coherence in esol learner texts. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Stroudsburg, PA, USA: ACL.