Computing Semantic Association: Comparing Spreading Activation and Spectral Association for Ontology Learning

Wohlgenannt, Gerhard; Belk, Stefan; Schett, Matthias

Published in:
Computing Semantic Association: Comparing Spreading Activation and Spectral Association for Ontology Learning

Published: 01/12/2013

Document Version:
Publisher's PDF, also known as Version of record

Document License:
Other

Computing Semantic Association: Comparing Spreading Activation and Spectral Association for Ontology Learning

Gerhard Wohlgenannt, Stefan Belk, and Matthias Schett

Vienna University of Economics and Business, Augasse 2-6, 1090 Wien, Austria
{gerhard.wohlgenannt,stefan.belk,matthias.schett}@wu.ac.at
http://www.wu.ac.at

Abstract. Spreading activation is a common method for searching semantic or neural networks, it iteratively propagates activation for one or more sources through a network – a process that is computationally intensive. Spectral association is a recent technique to approximate spreading activation in one go, and therefore provides very fast computation of activation levels. In this paper we evaluate the characteristics of spectral association as replacement for classic spreading activation in the domain of ontology learning. The evaluation focuses on run-time performance measures of our implementation of both methods for various network sizes. Furthermore, we investigate differences in output, i.e. the resulting ontologies, between spreading activation and spectral association. The experiments confirm an excessive speedup in the computation of activation levels, and also a fast calculation of the spectral association operator if using a variant we called *brute force*. The paper concludes with pros and cons and usage recommendations for the methods.

Keywords: spreading activation, spectral association, ontology learning

1 Introduction

Ontologies are the vocabulary and thereby the backbone of the Semantic Web. Constructing ontologies manually is an expensive and cumbersome process, and relies on highly specialized human labor [7]. (Semi-)automatic ontology learning from sources such as text supports the ontology engineer in his or her work.

Spreading activation is a method to search semantic networks, which can be used in ontology learning for example for finding semantically related concept candidates for existing concepts [14]. In the ontology learning system used as the foundation and test bed of this paper spreading activation is an essential tool used for the selection of domain-relevant concept candidates from a big semantic network as well as for positioning the new concepts in the ontology [16]. Spreading activation helps us to pick the most relevant concepts and associations from a vast number of evidence (candidate concepts and relations) generated from heterogeneous input sources.

In highly dynamic domains, and especially if the evolution of ontologies is of interest, new versions or updates of ontologies need to be generated in regular intervals. This makes the run-time performance for the ontology learning algorithms an important factor. Spreading activation is an iterative process and can be time-consuming to calculate, therefore Havasi et al. suggest a method called *spectral association* to compute the resulting activation levels after a number of steps of spreading activation in one step [11]. Spectral association approximates spreading activation using spectral decomposition of the concept matrix C, for details see Section 3.

This paper compares the two methods, i.e. the iterative spreading activation method and spectral association, in a specific environment for learning lightweight domain ontologies. We evaluate various aspects, including the run-time performance of both techniques and the impact on relevance of the resulting ontologies, discuss the implementation, and provide a general reflection of pros and cons of the methods observed as well as hints on when to apply which technique. The datasets used or domain is not important for the runtime of the evaluated algorithms – only the size of the semantic network.

Section 2 provides an overview of related work. The methods of spreading activation and spectral association are described in detail in Section 3. Section 4 introduces the ontology learning framework which applies the methods. Section 5 evaluates the run-time performance and other characteristics of the methods described earlier, and finally Section 6 summarizes the findings and gives an outlook on future work.

2 Related Work

This section starts with a presentation of a few selected ontology learning systems with respect to the question how evidence from algorithms or sources is integrated into the ontology. Text2Onto [3] uses a Probabilistic Ontology Model (POM) to aggregate results and represent the ontology. The results from various algorithms generate requests for changes of the POM; compared to the system for learning lightweight ontologies which this work is based on, Text2Onto aims at learning ontologies which are more expressive. Abraxas [18] takes quite a different approach, which is iterative and open-ended. The system identifies terms from a seed corpus and extracts ontological knowledge in the form of triples using lexico-syntactic patterns. Then it detects gaps in the collected ontological knowledge and tries to cover those with the help of external repositories such as the Web. KnowItAll [8] is not exactly an ontology learning system, but rather a system for the large scale extraction of facts. KnowItAll tests the plausibility of the candidate facts using pointwise mutual information (PMI) statistics computed by treating the Web as a massive corpus of text and thereby associates a probability with every fact. In their presentation of a generic architecture for ontology learning, Cimiano et al. stress the importance of methods of evidence integration especially when multiple and heterogeneous sources are combined, and state that a lot of further research is needed in that direction [2].

Spreading activation was first introduced by Collins et al. as a theory of human semantic processing [4]. Spreading activation is frequently used in information retrieval, Crestani provides a survey of spreading activation techniques on semantic networks in associative information retrieval [5]. The conclusions of the survey are positive, spreading activation is capable of providing good results.

Hasan proposes a system to apply spreading activation for information access within organizations [9]. The system integrates (i) documents, (ii) statistically derived information such as terms and entities extracted from those documents and (iii) a precise knowledge in form of an organizational ontology into a spreading activation network. User feedback on relevance of query results adapts the weights in the spreading activation network (learning).

Katifori et al. outline a framework which applies spreading activation over personal ontologies in the context of a personal interaction management system [13]. The goal of spreading activation is context inference depending on the users' recent actions and a populated personal information ontology to support user actions, for example generating suggestions when filling a Web form. The method is extended by augmenting the personal ontology with cached data from external repositories [6]. The selection of external data relates to the (spreading) activation level of entities already in the personal ontology or cached data.

Spectral association is an approximation technique for spreading activation networks, first presented in [11]. The paper describes the Colorizer application, which hypothesizes color values that represent given words and sentences. The common sense reasoning application determines the colors depending on physical descriptions of objects and emotional connotations. Spectral association interpolates colors for unknown concepts based on semantic relatedness to (in terms of color) known concepts. To ease computational complexity, Colorizer uses the spectral association approximation as a measure of semantic relatedness.

Spectral association is a variant of the AnalogySpace representation [15]. AnalogySpace addresses the problem of reasoning over large common sense knowledge bases with the characteristics of noisy and subjective data. The goal is to find rough conclusions based on similarities and tendencies, as traditional proof procedures are not feasible in such an environment. AnalogySpace forms analogical closures of a semantic network through dimensionality reduction.

A number of tools already utilize spectral association. Sentic Corner [1] is an application that dynamically collects audio, video, images and text related to and suitable for a user's current mood and displays this content on a multifaceted classification Website. The system detects user mood via the analysis of semantics and sentics on the user's microblog (e.g. Twitter) postings. Spectral association is one of the fundamental methods used in the system, it generates semantically related concepts from the descriptions of music, films, images and text itself.

The Glass Infrastructure [10] allows the discovery of latent connections between people, projects, and ideas in an organisation. The system uses spectral association to generate a "semantic space" from project information, where closeness in the space signifies similarity of people, projects and ideas.

3 Methods

As this paper focuses on comparing spreading activation and spectral association, this section will describe them in some detail.

Spreading activation is a technique for searching associative, neural and semantic networks. The method requires a network structure with numeric or discrete relations between the network nodes. In the beginning the activation level of all nodes in the network is set to zero. The process starts by labeling source nodes, that is setting the activation level of one or more nodes to a value higher than the firing threshold (F). Every unfired node with an activation level $> F$ fires to all its connected nodes, the energy propagated results from a multiplication of the energy of the source, the weight of the connection, and the decay factor D. In further iterations unfired nodes that received activation in the previous step will fire to their neighbors. The lower the decay factor D, the less activation is spread to nodes further away from the original source nodes. The process terminates when no more unfired nodes (with activation above the firing threshold F) exist in the network. As a result of the process, the activation levels of all nodes in the network give measures of association to the source nodes, i.e. the higher the activation level of a node in the network at the end, the stronger the association to the source node(s).

Equation 1 outlines the activation level adaption of a single node A_j when receiving energy from node A_i via a connection with weight $W_{i,j}$, reduced by the static decay factor D. Obviously, this energy accumulation is applied for every incoming (and firing) node of A_j.

$$A_j = A_j + (A_i \cdot W_{i,j} \cdot D) \tag{1}$$

As already mentioned, spreading activation can be time-consuming to calculate [11]. Spectral association approximates many steps of spreading activation. The spectral association method starts with the transformation of the spreading activation network into a square symmetric matrix C of concepts. The rows and columns of C are labeled with the concepts from the network, and the values in the matrix represent the relation strength between the concepts. The relation strength of a concept to itself is 1. If there was no connection in the spreading activation network between two concepts the value 0 results. Step two is to scale rows and columns to unit vectors. Applying C to a vector of activations (of one or more concepts) yields the result of one round of spreading activation. The operator e^C to simulate any number of spreading activation rounds (with diminishing returns) is calculated by Equation 2, see [11]:

$$e^C = 1 + C + \frac{C^2}{2!} + \frac{C^3}{3!} + \dots \tag{2}$$

As C is square symmetric, it can be decomposed to $C = V \Lambda V^t$. In this eigendecomposition, V is the orthogonal real matrix of eigenvectors, whereas Λ is the diagonal matrix of eigenvalues. We can raise this expression to any power,

and cancel anything but the power of Λ. What follows is that

$$e^C \approx V e^{\Lambda} V^t \tag{3}$$

To further ease computation, one can apply *dimensionality reduction* [11], i.e. keeping only the largest eigenvalues and their corresponding eigenvectors.

4 The Ontology Learning System

This section describes the system for learning domain-specific ontologies (T-box) underlying the work presented in this paper. The framework evolved since 2005 and has been presented in various publications. The original system [14] learns from domain text only and already includes spreading activation as the major building block to integrate evidence. Weichselbraun et al. evaluate information from social media as additional evidence source [16] and present novel methods for learning non-taxonomic relations [17]. Finally, Wohlgenannt et al. discusses data structures and algorithms for the fine-grained optimization of the system from feedback collected with games with a purpose [19].

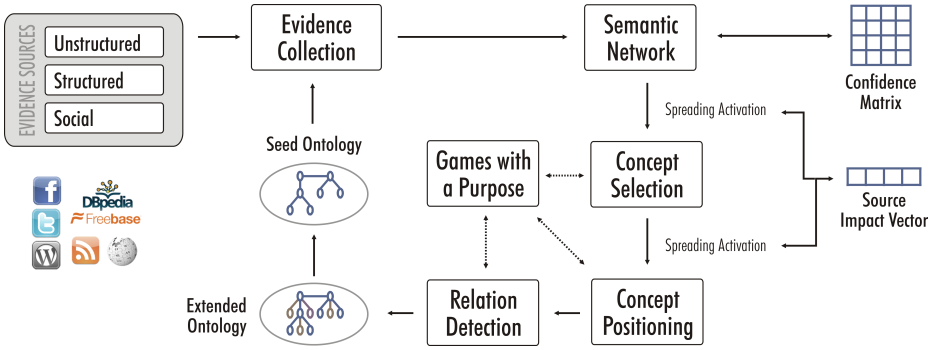

Fig. 1. Ontology Learning System Architecture Diagram [19].

The description given here is limited to a basic overview of the system needed to better understand the remainder of the paper. Figure 1 gives a graphical illustration of the major building blocks. The initial input to the process is a typically very small seed ontology (a few concepts and relations) in a particular domain. The system starts by collecting evidence for new concepts and relations in heterogeneous sources, i.e. domain text, social sources and structured sources (e.g. WordNet or online ontologies). A semantic network stores the gathered data. This semantic network is then transformed into a spreading activation network, from which new concept candidates are computed. Another round of spreading activation positions each of the new concepts in the seed ontology. Games with a purpose verify the relevance of new concepts and their position, as

well as help to optimize the system performance. Finally the extended ontology acts as new seed ontology for the next step of ontology extension. After a preconfigured number of extension steps the system halts.

So what is spreading activation actually used for? The evidence acquisition phase collects a large number of concept candidates and relations to the seed concepts, the number of candidate terms can easily exceed a few thousand. The system includes a big number of so-called *evidence sources*, for example co-occurring keywords in domain text of US media, UK media, related tags from twitter, disambiguated hyper-/hyponyms from WordNet, and many others. Each of the evidence sources generates a number of new candidate terms (in this context we use *term* and *concept* synonymously) for any seed concept – all this information gets collected in the *semantic network*. A transformation algorithm creates the spreading activation network from the semantic network, the evidence source which suggested the relation influences the weight of the link. Spreading activation detects the most relevant n concepts (where $n = 25$ for example) from all the candidates. Roughly, candidates which are supported by different sources and by sources with a higher link weight ("source impact") are more likely to be selected.

Spreading activation is further used to position the selected concepts with respect to the seed ontology. For this purpose, the system reverses the activation flow in the network, and for every new concept determines the seed concept with the strongest association.

Figure 2 shows an extended ontology generated by the ontology learning system. Starting from the seed ontology (yellow), the first round of extension generated and positioned new concepts (light-green, "stage one"). Further rounds, namely stage two (green) and three (dark-green), were used to expand the ontology.

In our implementation of the spreading activation algorithm we use real valued weights. The weights are normalized in the range [0.0, 1.0]. We experimented with a number of decay factor values, and decided on $D = 0.3$. As we are looking for concepts closely related to and in close vicinity of the seed concept, a small decay factor is appropriate. No firing threshold is used, i.e. $F = 0.0$.

The implementation of the spectral association approximation of spreading activation starts with building the square symmetric concept matrix. Next a normalization step takes place. Havasi et al. propose to use unit vectors for rows and columns [11], but this led to increasing returns in Equation 2 when increasing n in $\frac{C^n}{n!}$ – which is not the expected behavior. Using the *determinant* of the matrix instead for normalization gave results as expected.

We calculate the operator e^C in two ways. The first variant uses spectral decomposition (eigendecomposition) to ease the approximation of e^C, in line with the description in Section 3 of the seminal paper [11]. This variant is referenced as *spectral association* or *SPECTRAL* in the evaluation section (Section 5). For the computation of eigenvectors and eigenvalues we use NumPy's `linalg.eigh` module [12], which is geared towards the generation of eigenvectors for symmetric

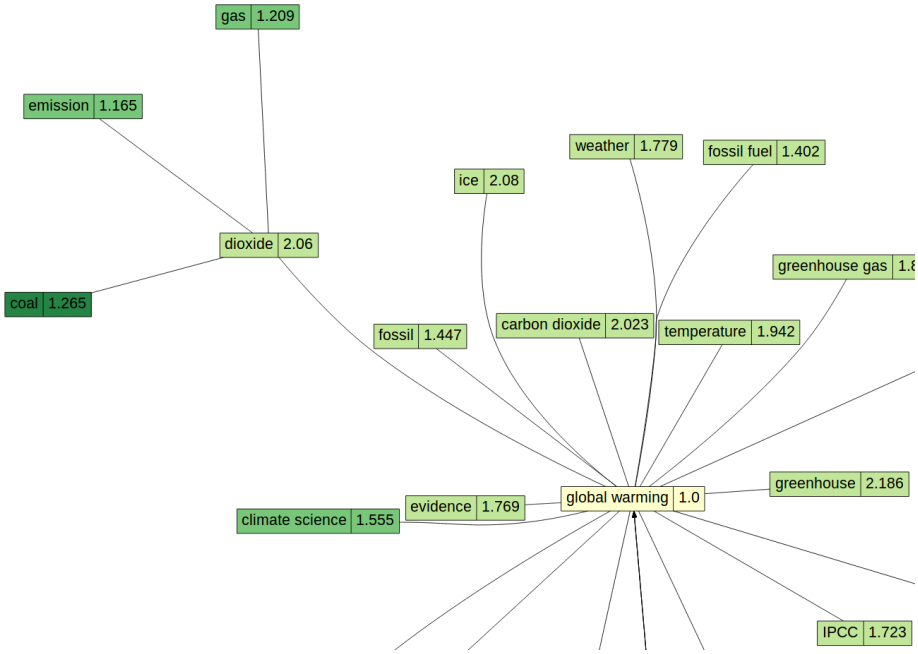

Fig. 2. An extended ontology (clipped) – concept labels and relations

matrices. NumPy[1] is a package for scientific computing in Python. The second variant, called *brute force* or *BRUTE* just calculates e^C according to Equation 2 – without the use of eigendecomposition. This makes the matrix multiplications more complicated, but saves the computation of eigenvectors. We use *brute force* (i) to validate the correct implementation of the *spectral association* method, and (ii) as a baseline when evaluating the run-time characteristics of spectral association.

Finally, as soon as e^C is computed, it can be used to simulate the activation of nodes simply by multiplying (using the dot-product) e^C with a concept vector where the values for the concepts to activate are set to 1, else 0.

The current version of the spectral association method can be found as an open source package on the Web[2] – free for use and modification. It is written in the Python programming language and uses `numpy` for all matrix operations. The most important parameter for run-time performance tuning is `APPROX_DEPTH`. In Equation 2, the implementation calculates C as far as the power of `APPROX_DEPTH`. We currently use the value of 10 – which proved to be more than sufficient in our experiments, as the factor gets very small (close to zero) with higher powers. The same parameter is used in the approximation of e^Λ, too.

[1] http://www.numpy.org
[2] `http://wwwai.wu.ac.at/~wohlg/spectral_association`

5 Evaluation

This section provides an extensive evaluation of the methods described. As the purpose of spectral association is to approximate and speed up the spreading activation process, the evaluation focuses on a comparison of run-time characteristics. Furthermore we check if there are significant differences in the resulting ontologies regarding quality of concepts or quality of concept positioning.

Table 1 presents timings for the ontology extension phase, i.e. the total runtime of the step of classic spreading activation (`SPREADING`), and the two variants of spectral association: with spectral decomposition (`SPECTRAL`) and brute force (`BRUTE`). `concepts` refers to the number of nodes in the spreading activation network, which also determines the size of the concept matrix C in spectral association. The `connections` are the links between concepts in the network. The table includes *average* data over multiple ontology generation runs for each of the three ontology extension stages (`Avg stage-1/2/3`). Furthermore, it depicts the single stage with the most concepts and connections (`biggest`) and a spectral association run which uses dimensionality reduction (`With DR`).

Run	concepts	connections	SPREADING	SPECTRAL	BRUTE
Avg stage-1	2495	3303	00:00:10 (1)	00:05:56	00:00:11
Avg stage-2	3843	9054	00:40:56 (91)	00:30:05	00:00:57
Avg stage-3	6101	18655	00:38:47 (86)	01:22:40	00:01:54
Biggest	6842	22342	00:44:35 (109)	01:43:46	00:02:35
With DR	6842	22342	–	00:35:33	–

Table 1. Run-times of the ontology extension step depending on the number of `concepts` and the number of `connections` for spreading activation and the two variants of spectral association.

It was very surprising to see that the *brute force* method performed best by far overall, for networks with > 5000 connections 20-40 times faster than the other two. Classic spreading activation sometimes outperformed spectral association, especially for small networks, and if no dimensionality reduction was applied.

We performed a thorough analysis to investigate *where time is spent*:

- *SPREADING activation*: As expected, almost all of the computing time is consumed by applying the activation algorithm to the network in concept detection and concept positioning.
- *BRUTE force* spends most of its time (about 70%) generating the operator e^C according to Equation 2.
- *SPECTRAL association* uses almost all of its time in the generation of e^C. In doing so, NumPy consumes about 33% for generating the eigenvectors and -values. The matrix multiplication in Equation 3 consumes the remaining computation time. This single matrix multiplication is surprisingly costly, keeping in mind that we have around 10 matrix multiplications (with matrices of same size) in Equation 2 when computing e^C *brute force* – depending

on the `APPROX_DEPTH`. Additional investigation showed that the matrices in Equation 3 are very dense, and that the concept matrix C in Equation 2 is typically sparse – an important point further covered below.

Another surprising observation is the huge difference between *Avg stage-1* (2672 concepts, 11 seconds) and *Avg stage-2* (3154 concepts, 26 minutes) for the method *spreading activation*. This is caused by two factors: (i) In stage-1 the network depth is very low (most nodes are connected to the seed concepts directly), whereas in stage-2 the network depth increases – which results in more steps of activation propagation. (ii) The number of activation processes differs. In stage-1 the system just needs to activate the seed concepts, while in stage-2 it positions the new concepts from stage-1 and activates the network. This results in about 60-110 activations, the exact number is given in parentheses in Table 1. A single activation in stage-2 and stage-3 has a runtime of about 20-30 seconds.

The eigenvector variant of spectral association can be approximated further (and thereby sped up) by *dimensionality reduction* (DR, see Section 3). Our implementation has an (optional) parameter to set the requested number of dimensions, we used a value of 50 in the experiments. As can be seen in Table 1, line `With DR`, applying DR reduces runtime to about 35%. The 35% largely result from calling NumPy for the calculation of eigenvectors (see above), so this variant is very effective to speed up the process. The use of DR had no impact on the resulting ontologies, the approximation works as expected.

Table 2 depicts memory usage for the three methods depending on the network size. The presented data reflects the maximum amount of memory allocated during runtime of the process.

Run	concepts	connections	SPREADING	SPECTRAL	BRUTE
stage-1	2672	3531	0.5 GB	0.4 GB	0.4 GB
stage-2	3154	12243	1.0 GB	2.0 GB	1.7 GB
stage-3	6842	22342	1.2 GB	3.3 GB	2.7 GB

Table 2. Maximum memory usage of the ontology extension step depending on the stage and algorithm used.

We did not focus on optimizing memory usage at all, the goal of Table 2 is to give rough comparison of the methods, and to observe scaling regarding memory consumption. As expected, memory usage tends to scale linearly with the number of concepts and connections. In variant *SPECTRAL*, memory usage is highest during NumPy's computation of eigenvectors and -values. If required, there will be a number of options to decrease memory usage, for example experimenting with Numpy's sparse matrix types or specific optimizations regarding memory in the Python code. The machine we ran the experiments on had enough memory to prevent any swapping to disk.

Complementing the evaluation of run-time performance, we also compared the quality of results. Spectral association is intended as approximation of spread-

ing activation, so results should roughly be the same, although there are various parameter settings available. We did three ontology extension runs for March, April and May 2013 which extended the seed ontology with 25 candidate concepts in each of the three extension iterations, summing up to 225 candidate concepts. Manual evaluation by domain experts showed that there was no significant difference in the relevance of candidate concepts. The relevance was about 50% for all three methods (50% BRUTE, 50% SPECTRAL, 49% SPREADING). We also investigated the effect of the application of the three methods on the quality of concept positioning; no significant differences were found.

The experiments conducted suggest the following *pros and cons*, as well as *areas of application* for the tree methods:

- *Spreading activation*: Among the pros of spreading activation is the interpretability of the activation process, which can be traced easily, whereas spectral association is rather a black box model. Spreading activation provides parameters for tuning the activation process to fit ones specific needs (mainly with the firing threshold F and decay D), while we do not see a simple way how to apply these tuning parameters in the spectral association process. As stated above, the *depth* of the network, i.e. the number of propagation steps necessary, strongly affects runtime performance. The more propagation steps, the slower the process. Naturally, the number of propagation steps also depends on decay factor D and firing threshold F.
  The application of the method is advisable for small networks and situations in which runtime performance is not an issue, in the (unlikely) case of only one or a few activations needed, or if parameter tuning is required.
- *Variants of spectral association*: In contrast to spreading activation, in both spectral association variants the main factor is the generation of e^C, once this is done activation is very fast - it's just a simple $matrix \circ vector$ multiplication. This is a crucial point, activation processes can be done almost instantly with this method. If the activation process has to be further quickened, one can apply dimensionality reduction by using only the largest eigenvalues and their corresponding eigenvectors [11]. Dimensionality reduction also speeds up the the generation of e^C, in our experiments by a factor of approx. 3. However, it is not applicable in the *brute force* variant (as we do not compute eigenvectors).
  The sparser the concept matrix, i.e. the less connections between concepts, the faster the computation of e^C with the *brute force* method. We also tested with a dense matrix (almost all values $\neq 0$), in that case *brute force* is consistently slower (around factor 2) than *SPECTRAL* overall. But in a typical scenario with a sparse concept matrix the computations in Equation 2 are very efficient.
  - *Spectral association (with eigendecomposition)*: The variant allows the application of dimensionality reduction to further reduce and approximate e^C, and thereby enables even faster activation processes. It is well suited for very dense networks and if the runtime of activation is critical, and not so much the generation of e^C.

- *Brute force*: In our run-time analyses which have to be further confirmed in other environments and programming languages, the described *brute force* approach is surprisingly efficient. *Brute force* is preferable when the number of activations is limited and the time spent in both the generation of e^C as well as activations is critical – as in our ontology learning framework, where the method was performing best by far.

6 CONCLUSIONS

This paper compares a classic method for searching networks, i.e. spreading activation, with spectral association. Spectral association approximates the computationally intensive activation process using a matrix operator called e^C. The generation of this matrix via spectral decomposition is complex for large spreading activation nets, therefore spectral association is particularly well suited where e^C is used for many activation processes. In situations where instant results for an activation (e.g. in interactive applications) are needed, spectral association is the only option. We also tested a simpler way to generate the operator e^C, named *brute force*, which – under the circumstances we investigated – provides far superior runtime performance in generating e^C.

The main contributions of this paper are (i) extensively evaluating spreading activation vs. spectral association in the domain for ontology learning regarding various sizes of concept networks, (ii) showing that the *brute force* method is very efficient (20–40 times faster than spreading activation) in our experiments, (iii) the provision of an (open-source) implementation of spectral association in Python, (iv) giving hints under what circumstances to apply which method.

Future work will include experimenting with a wider range of network sizes and parameter settings, using other libraries for example to compute eigenvectors, and the application of other programming languages to verify the results.

Acknowledgements. The work presented in this paper was developed within DIVINE (www.weblyzard.com/divine), a project funded by the Austrian Ministry of Transport, Innovation & Technology (BMVIT) and the Austrian Research Promotion Agency (FFG) within FIT-IT (www.ffg.at/fit-it). The work has also been supported by uComp (www.ucomp.eu), a project in EU's ERA-NET CHIST-ERA programme.

References

1. Cambria, E., Hussain, A., Eckl, C.: Taking refuge in your personal sentic corner. In: Bandyopadhyay, S., Okumurra, M. (eds.) Proceedings of IJCNLP, Workshop on Sentiment Analysis where AI meets Psychology. pp. 35–43. Chiang Mai, Thailand (2011)
2. Cimiano, P., Mädche, A., Staab, S., Völker, J.: Ontology learning. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 245–267. International Handbooks Information System, Springer Berlin Heidelberg (2009)

3. Cimiano, P., Völker, J.: Text2onto. In: Montoyo, A., Muñoz, R., Métais, E. (eds.) NLDB. Lecture Notes in Computer Science, vol. 3513, pp. 227–238. Springer (2005)
4. Collins, A.M., Loftus, E.F.: A spreading-activation theory of semantic processing. Psychological Review 82(6), 407–428 (1975)
5. Crestani, F.: Application of spreading activation techniques in information retrieval. Artificial Intelligence Review 11(6), 453–482 (1997)
6. Dix, A.J., Katifori, A., Lepouras, G., Vassilakis, C., Shabir, N.: Spreading activation over ontology-based resources: from personal context to web scale reasoning. International Journal of Semantic Computing 4(1), 59–102 (2010)
7. Drumond, L., Girardi, R.: A survey of ontology learning procedures. In: de Freitas, F.L.G., Stuckenschmidt, H., Pinto, H.S., Malucelli, A., Corchoo, Ó. (eds.) Proceedings of the 3rd Workshop on Ontologies and their Applications (WONTO). CEUR Workshop Proceedings, vol. 427. CEUR-WS.org, Salvador, Brazil (October 2008)
8. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Web-scale information extraction in knowitall: (preliminary results). In: WWW '04: Proceedings of the 13th international conference on World Wide Web. pp. 100–110. ACM, New York, NY, USA (2004)
9. Hasan, M.M.: A spreading activation framework for ontology-enhanced adaptive information access within organisations. In: van Elst, L., Dignum, V., Abecker, A. (eds.) AMKM. Lecture Notes in Computer Science, vol. 2926, pp. 288–296. Springer (2003)
10. Havasi, C., Borovoy, R., Kizelshteyn, B., Ypodimatopoulos, P., Ferguson, J., Holtzman, H., Lippman, A., Schultz, D., Blackshaw, M., Elliott, G.T.: The glass infrastructure: Using common sense to create a dynamic, place-based social information system. AI Magazine 33(2), 91–102 (2012)
11. Havasi, C., Speer, R., Holmgren, J.: Automated color selection using semantic knowledge. In: AAAI Fall Symposium Series. Arlington, Texas (2010)
12. Jones, E., Oliphant, T., Peterson, P., et al.: SciPy: Open source scientific tools for Python (2001–), `http://www.scipy.org/`
13. Katifori, A., Vassilakis, C., Dix, A.J.: Ontologies and the brain: Using spreading activation through ontologies to support personal interaction. Cognitive Systems Research 11(1), 25–41 (2010)
14. Liu, W., Weichselbraun, A., Scharl, A., Chang, E.: Semi-automatic ontology extension using spreading activation. Journal of Universal Knowledge Management 0(1), 50–58 (2005)
15. Speer, R., Havasi, C., Lieberman, H.: Analogyspace: Reducing the dimensionality of common sense knowledge. In: Fox, D., Gomes, C.P. (eds.) AAAI. pp. 548–553. AAAI Press (2008)
16. Weichselbraun, A., Wohlgenannt, G., Scharl, A.: Augmenting lightweight domain ontologies with social evidence sources. In: Tjoa, A.M., Wagner, R.R. (eds.) 9th International Workshop on Web Semantics, 21st International Conference on Database and Expert Systems Applications (DEXA 2010). pp. 193–197. IEEE Computer Society Press, Bilbao, Spain (August 2010)
17. Weichselbraun, A., Wohlgenannt, G., Scharl, A.: Refining non-taxonomic relation labels with external structured data to support ontology learning. Data & Knowledge Engineering 69(8), 763–778 (2010)
18. Wilks, Y., Brewster, C.: Natural language processing as a foundation of the semantic web. Foundations and Trends in Web Science 1(3-4), 199–327 (2009)
19. Wohlgenannt, G., Weichselbraun, A., Scharl, A., Sabou, M.: Dynamic integration of multiple evidence sources for ontology learning. Journal of Information and Data Management (JIDM) 3(3), 243–254 (2012)