JoBimViz: A Web-based Visualization for Graph-based Distributional Semantic Models

Eugen Ruppert and **Manuel Kaufmann** and **Martin Riedl** and **Chris Biemann**
FG Language Technology
Computer Science Department, Technische Universität Darmstadt,
Hochschulsstrasse 10, D-62489 Darmstadt, Germany
{eugen.ruppert,riedl,biem}@cs.tu-darmstadt.de, mtk@kisad.de

Abstract

This paper introduces a web-based visualization framework for graph-based distributional semantic models. The visualization supports a wide range of data structures, including term similarities, similarities of contexts, support of multiword expressions, sense clusters for terms and sense labels. In contrast to other browsers of semantic resources, our visualization accepts input sentences, which are subsequently processed with language-independent or language-dependent ways to compute term-context representations. Our web demonstrator currently contains models for multiple languages, based on different preprocessing such as dependency parsing and n-gram context representations. These models can be accessed from a database, the web interface and via a RESTful API. The latter facilitates the quick integration of such models in research prototypes.

1 Introduction

Statistical semantics has emerged as a field of computational linguistics, aiming at automatically computing semantic representations for words, sentences and phrases from (large) corpora. While early approaches to distributional semantics were split into symbolic, graph-based approaches (Lin, 1998) and vector-based approaches (Schütze, 1993), recent trends have mainly concentrated on optimizing the representation of word meanings in vector spaces and how these account for compositionality (cf. Baroni and Lenci (2010); Turney and Pantel (2010)).

While dense vector representations, obtained by singular value decomposition (cf. Rapp (2002)) or neural embeddings (Mikolov et al., 2010), have gained popularity due to successes in modelling semantic and relational similarity, we propose to revisit graph-based approaches to distributional semantics in the tradition of Lin (1998), Curran (2002) and Biemann and Riedl (2013) – at least as an additional alternative – for the following reasons:

- (dense) vector representations are not interpretable, thus it cannot be traced why two terms are similar
- vectors do not make word senses explicit, but represent ambiguous words as a mix of their senses
- graph-based models can be straightforwardly structured and extended, e.g. to represent taxonomic or other relationships

In this demonstration paper, we describe JoBimViz, a visualization and interactive demonstrator for graph-based models of distributional semantics. Our models comprise similarities between terms (a.k.a. distributional thesaurus) and multiword units, similarities between context features, clustered word senses and their labeling with taxonomic is-a relations. The demonstrator transforms input sentences into a term-context representation and allows to browse parts of the underlying model relevant to the sentence at hand. Requests are handled through a RESTful API, which allows to use all available information in custom prototypes via HTTP requests. The demonstrator, as well as the computation of the models, is fully available open source under a permissive license.

2 Related Work

Aside from providing a convenient lookup for human users, the visualization of semantic models provides an important tool for debugging models visually. Additionally, prototyping of semantic applications based on these models is substantially accelerated.

VISUALSYNONYMS[1] and THESAURUS.COM[2] offer lookup possibilities to retrieve various information for input words. Users can view information, like WordNet or Wikipedia definitions and related words (synonyms, hypernyms, etc.). BABELNET[3] (Navigli and Ponzetto, 2012) uses such information to compile multilingual definitions and translations for input words. Here, the words are differentiated by senses, with taxonomical labels. BABELNET offers a SPARQL endpoint and APIs for web access.

SERELEX[4] (Panchenko et al., 2013) is a graphical thesaurus viewer. Users can enter a term for different languages and retrieve related words. The similarity graph displays the similarity links between similar items ('secondary links'). The items can be expanded for a denser graph. The input terms map to nominal phrases, allowing the interface to display short definitions and disambiguations from Wikipedia. An API with JSON output for similar words is provided.

SKETCH ENGINE[5] (Kilgarriff et al., 2004) offers access to pre-processed corpora. For each corpus, the user can view concordances and similar terms (thesaurus) for a given term. SKETCH ENGINE also features statistical information, like word frequency and co-occurrence counts. Furthermore it shows meta information for a corpus. One drawback of the SKETCH ENGINE is that the tools and the models are not freely available.

NETSPEAK[6] (Stein et al., 2010) is a search engine for words in context. Access is possible via a graphical UI, a RESTful interface and a Java API. Users can enter wildcards and other meta symbols into the input phrases and thus retrieve all the words and phrases that occur in a given context. The information is displayed with corpus statistics.

A novel aspect of JOBIMVIZ is that it incorporates several different aspects of symbolic methods (distributional thesaurus, context feature scores, sense clusters), and all of these methods are derived from the input corpus alone, without relying on external resources, which are not available for every language/domain. Furthermore, we provide domain-based sense clusterings with is-a labels (cf. Figure 4), which is not performed by the other discussed tools.

Our interface features a generic interactive visualization that is adaptable to different kinds of parses and also handles multiword units in the visualization. All of this information is made freely available by the API, enabling rapid prototyping techniques. To our knowledge, the presented demonstrator is the only online tool that combines technical accessibility (open source project, open API) with rich, flexible preprocessing options (cf. Section 3) and graph-based, structured semantic models that contain context similarities.

3 Computation of distributional models

The visualization is based on distributional models computed with the JoBimText framework (Biemann and Riedl, 2013)[7]; however it can also be used for other semantic models of similar structure. One of the major components of the framework is a method for the computation of distributional thesauri. This method consists of two steps: a holing operation and a similarity computation.

The holing operation processes text and yields a representation consisting of jos and bims. Jos and bims are normally instantiated by a term (jo) and its context features (bims), but the definition extends to arbitrary splits of the input perception that mutually characterize each other distributionally. The holing operation executes a preprocessing pipeline that can be as simple as text segmentation or complex like POS tagging and dependency parsing, depending on the complexity of the context features. As the preprocessing is defined in UIMA (Ferrucci and Lally, 2004) pipeline descriptors, it is straightforward to exchange components or define new preprocessing operations. Using this processed and annotated text, the holing annotator generates the term–feature representation of the input text, e.g. by using the neighboring words ('trigram holing') or dependency paths

[1] http://www.visualsynonyms.com
[2] http://www.thesaurus.com
[3] http://babelnet.org/
[4] http://serelex.org/
[5] http://www.sketchengine.co.uk/
[6] http://www.netspeak.org/

[7] The framework is available under the permissive ASL 2.0 license at http://sf.net/p/jobimtext.

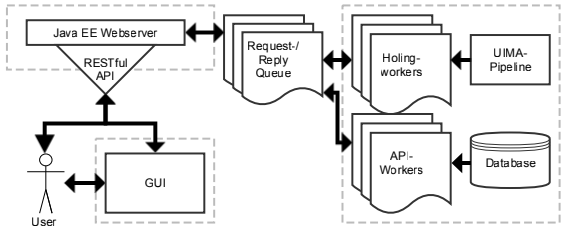

Figure 1: Architecture of JoBimViz, with the three components GUI, Web Server and Workers.

between terms ('dependency holing'). A graphical example is given in Figure 5, where the holing operation yields four context features for the term `example#NN`. Different term representations are possible, like surface text, lemma, lemma+POS and also multiword expressions. This flexibility allows, on the one hand, domain- and language-independent similarity computations using general holing operations, while on the other hand allowing complex, language-specific processing in the holing operations for higher quality models (e.g. using parsing and lemmatization).

The second part consists of the similarity computation, which relies on MapReduce (Dean and Ghemawat, 2004) for scalability and employs efficient pruning strategies to keep runtime feasible even for very large corpora. After the computation of similarities, sense clusters for each term are computed using Chinese Whispers graph clustering (Biemann, 2006), as described in Biemann et al. (2013). Furthermore, the sense clusters are also labeled with hypernyms (is-a relations) derived from Hearst patterns (Hearst, 1992), implemented using the UIMA Ruta (Kluegl et al., 2014) pattern matching engine[8].

4 Web-based Demonstrator

4.1 Architecture and Technology

The architecture of JoBimViz consists of three main components (see Figure 1). The central element is a Java EE based web server, which provides a RESTful interface (Fielding, 2000) for accessing API resources, such as sentence holing operations and the distributional models.

To handle many parallel requests for long running holing operations like dependency parsing, we use an Apache ActiveMQ[9] based request/reply queue. All requests to the web server are stored in the queue and processed by one of the worker processes, which can be distributed on different machines and handle the requests in parallel. These workers form the second component of our system. The holing workers execute the UIMA pipelines that define the holing operations. Usage of UIMA descriptors provides great flexibility, since one type of workers can run every holing operation. Every model defines a custom UIMA pipeline to ensure the same holing operation for the input and the model. To speed up the holing operation we cache frequently queried holing outputs into the in-memory database Redis[10]. Requests to the API are processed by another type of workers, which retrieve the relevant data from the models database.

The third component of the software is a HTML5-based GUI with an overall layout based on the Bootstrap[11] framework. The front-end uses Ajax requests to connect to the RESTful interface of the web server and retrieves responses in the JSON format. The received data is processed by a Javascript application, which uses D3.js (Bostock et al., 2011) to visualize the graphs.

4.2 Visualization

In the demonstrator, users can enter sentences or phrases, run different holing operations on the input and browse distributional models. The demonstrator can be accessed online[12].

Figure 2 shows the application layout. First, the user can decide on the model, which consists of a corpus and a holing operation, and select it via the dropdown holing operation selector (3). Then, the user enters a sentence in the text field (1) and activates the processing by clicking the *Parse* button (2). The holing output is presented as a graph (4) with marked terms and context features. Other views are available in tab bar at the top (5). To retrieve term similarities, a term in the displayed sentence can be selected (4a), activating the information boxes (6, 7, 8, 9). Context similarities can be viewed by selecting the corresponding feature arc (4b). The frequency of the selected term/feature is presented on the top right (6). Similar items are displayed in the first box in the lower pane (7). Similarity scores between

[8] `https://uima.apache.org/ruta.html`
[9] `http://activemq.apache.org/`
[10] `http://redis.io`
[11] `http://www.getbootstrap.com/`
[12] `http://goo.gl/V2ZEly`

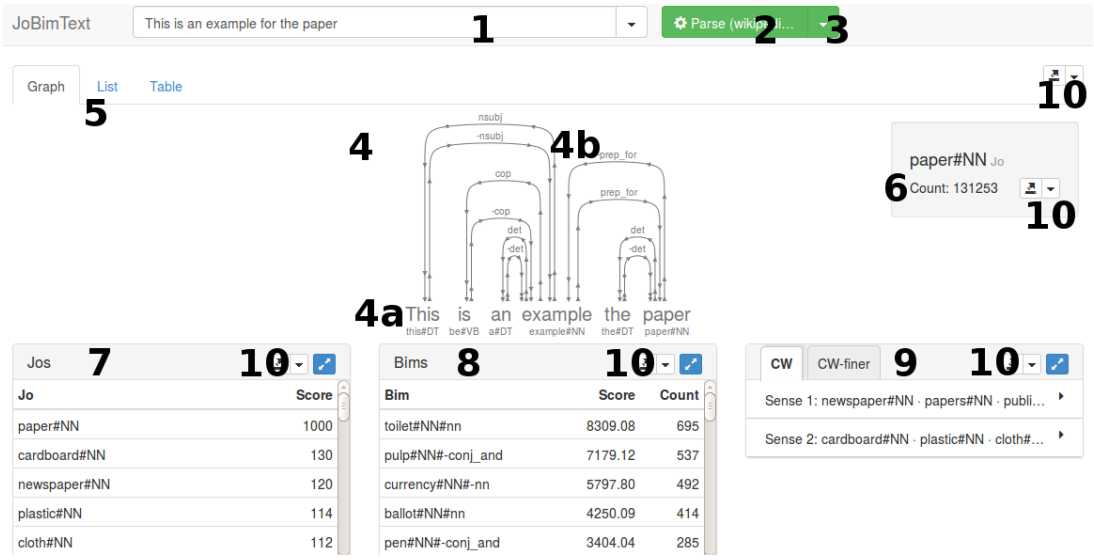

Figure 2: Overview of the visualization with a collapsed dependency parse of the input sentence *This is an example for the paper*; the selected term is *paper#NN*.

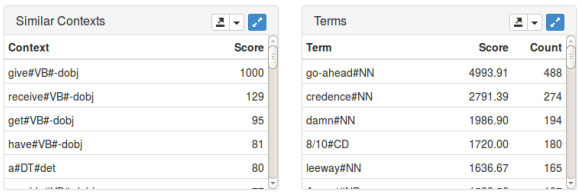

Figure 3: Similar bims (left) and most significant jos (right) for the dependency parse context `give#VB-dobj` (direct object of *give*).

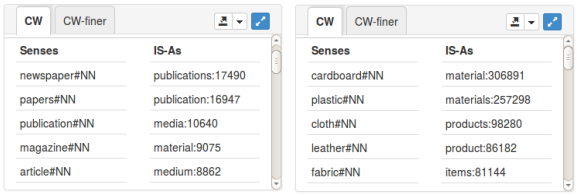

Figure 4: Different word senses for *paper*, with hypernym terms (sense 1 *paper:publication*, sense 2 *paper:material*), as accessed from field (9) in Figure 2. The tabs "CW/CW-finer" provide access to different sense clustering inventories, e.g. with a different granularity.

the selected and similar terms are shown, including self-similarities as an upper limit for similarity of the selected item.

The most relevant context features for the term are displayed with the significance score and their term-feature count in the corpus (8). When selecting a context feature, the most relevant terms for a context feature are shown (cf. Figure 3). For terms, there is a box displaying sense clusters (9). These are often available in different granularities to match the application requirements (e.g. 'CW' or 'CW-finer')[13]. When a user selects a sense cluster, a list of related terms for the selected cluster and a list of hypernyms (is-a relations) with frequency scores are displayed (cf. Figure 4). Buttons for API calls are displayed for all data display in the GUI (10). This enables users to get comfortable with the models and the API before deploying it in an application. The buttons feature selectors for different output data format options, i.e. TSV, XML, JSON and RDF. For the boxes with list content, there is a 'maximize' button next to the API button that brings up a screen-filling overlay.

[13]Here, 'CW' is referring to sense clusters computed with Chinese Whispers (Biemann, 2006).

4.2.1 Sentence Holing Operations

For the graphical representation of holing operations, the web demo offers views for single terms (Figure 5 and 6) as well as support for n-grams (Figure 7). Figure 5 shows the tree representation for a dependency parser using collapsed dependencies. Figure 6 exemplifies a holing operation considering the left and right neighboring words as one context feature ('trigram holing'). Figure 7 shows the result of the same holing operation, applied for n-grams, where several different possible left and right multiword items can be selected as context. Here, the demonstrator identified mul-

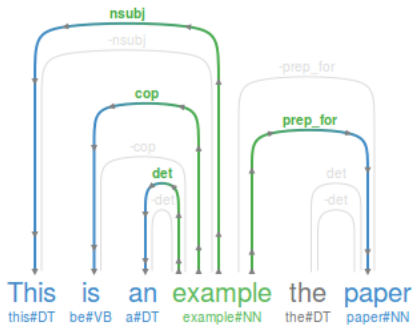

Figure 5: Collapsed dependency (de Marneffe et al., 2006) holing result for *This is an example for the paper*; the preposition *for* is collapsed into the `prep_for` dependency.

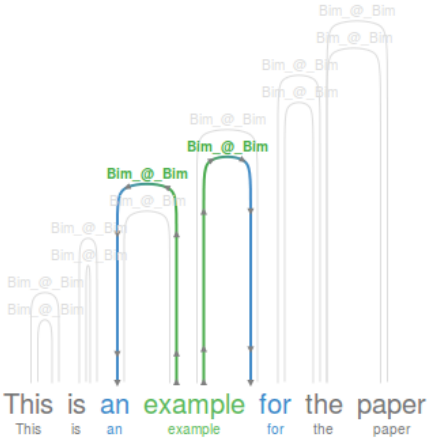

Figure 6: Trigram holing (unigram) result for *This is an example for the paper*.

tiword expressions that are present in the corresponding distributional model (*acute lymphoblastic leukemia, lymphoblastic leukemia cells* and *human bones*). These expressions can be selected like single word items. Furthermore, there is a filtering function for n-grams, where users can refine the display of n-grams by selecting the desired n-gram lengths.

4.2.2 Model Access

The demonstrator features a selection of models for different languages (currently: German, English, Hindi, Bengali) and different domains, like news, encyclopedia or medical domain. Besides term similarities, typical context features and sense clusters are also part of these models. Distributional similarities for context features can be viewed as well. By selecting an arc that represents the feature relation, the user can view similar features in the GUI. In Figure 3, the context features

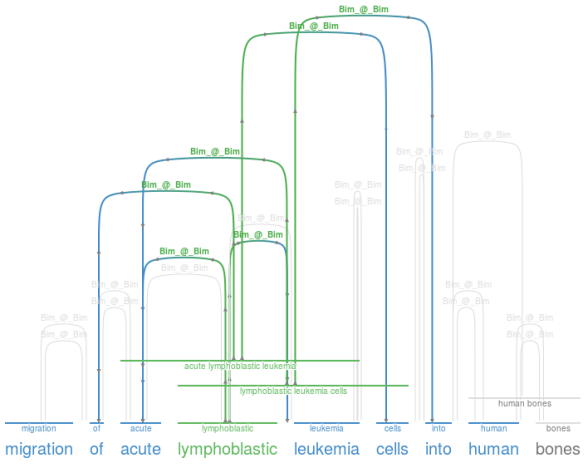

Figure 7: Trigram holing (n-gram) result for *migration of acute lymphoblastic leukemia cells into human bones* with display of multiwords that are part of the model.

that are most similar to `give#VB#-dobj` (direct object of verb *give*) are displayed. Here, the most significant words for a feature are shown. To our knowledge, we are the first ones to explicitly provide similarities of contexts in distributional semantic models.

Holing operations and models can be accessed via an open RESTful API. The access URIs contain the model identifier (consisting of dataset and holing operation), the desired method, like sentence holing or similar terms, and the input sentence, term or context feature. The distributional project also features a Java API to access models via the web-based API[14].

5 Conclusion and Future Work

In this paper we have introduced a new web-based tool that can be used to browse and to access graph-based semantic models based on distributional semantics. In its current form, it can display data from a distributional thesaurus, similarities of context features, sense clusters labeled with taxonomic relations, and provides the display of multiword expressions. Additionally, it provides the functionality to transform sentences into term–context representations. The web demo can give a first impression for people who are interested in the JoBimText framework for distributional semantics. Providing a RESTful interface for accessing all information with state-

[14]For an overview of available API methods, see `http://goo.gl/l6K6Gu`.

less requests allows for an easy integration into prototypes. The RESTful API can also be accessed using our Java API, which also can access other back-ends such as on-disk and in-memory databases. The complete project is available under the permissive Apache ASL 2.0 license and models for several languages are available for download[15].

Whereas at the moment similar terms are globally ranked, we will add visualization support for a contextualization method, in order to rank similar terms regarding their context within the sentence. Furthermore, we are working on incorporating more complex pre-processing for the holing operation in the visualization, e.g. aggregating context features over co-reference chains, as well as relation extraction and frame-semantic parsing for term–context representations.

Acknowledgments

This work has been supported by the German Federal Ministry of Education and Research (BMBF) within the context of the Software Campus project *LiCoRes* under grant No. 01IS12054 and by an IBM Shared University Research award.

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

Chris Biemann and Martin Riedl. 2013. Text: Now in 2D! a framework for lexical expansion with contextual similarity. *Journal of Language Modelling*, 1(1):55–95.

Chris Biemann, Bonaventura Coppola, Michael R. Glass, Alfio Gliozzo, Matthew Hatem, and Martin Riedl. 2013. JoBimText Visualizer: A graph-based approach to contextualizing distributional similarity. In *Proc. TextGraphs 2013*, pages 6–10, Seattle, Washington, USA.

Chris Biemann. 2006. Chinese Whispers – an efficient graph clustering algorithm and its application to natural language processing problems. In *Proc. TextGraphs 2006*, pages 73–80, New York City, NY, USA.

Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. 2011. D3: Data-driven documents. *IEEE Transactions on Visualization & Computer Graphics*, 17(12):2301–2309.

James R. Curran. 2002. Ensemble methods for automatic thesaurus extraction. In *Proc. EMNLP'02/ACL-2002*, pages 222–229, Philadelphia, PA, USA.

[15]Selection of models available under `http://sf.net/projects/jobimtext/files/data/models/`.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proc. LREC-2006*, pages 449–454, Genova, Italy.

Jeffrey Dean and Sanjay Ghemawat. 2004. MapReduce: Simplified Data Processing on Large Clusters. In *Proc. OSDI '04*, pages 137–150, San Francisco, CA, USA.

David Ferrucci and Adam Lally. 2004. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering*, 10(3-4):327–348.

Roy Thomas Fielding. 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irivne.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. COLING-1992*, pages 539–545, Nantes, France.

Adam Kilgarriff, Pavel Rychlý, Pavel Smrž, and David Tugwell. 2004. The Sketch Engine. In *Proc. EURALEX 2004*, pages 105–116, Lorient, France.

Peter Kluegl, Martin Toepfer, Philip-Daniel Beck, Georg Fette, and Frank Puppe. 2014. UIMA Ruta: Rapid development of rule-based information extraction applications. *Natural Language Engineering*.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proc. COLING-98*, pages 768–774, Montréal, Quebec, Canada.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proc. INTERSPEECH 2010*, pages 1045–1048, Makuhari, Chiba, Japan.

Roberto Navigli and Simone P. Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Alexander Panchenko, Pavel Romanov, Olga Morozova, Hubert Naets, Andrey Philippovich, Alexey Romanov, and Fairon Cédrick. 2013. Serelex: Search and visualization of semantically related words. In *Proc. ECIR 2013*, pages 837–840, Moscow, Russia.

Reinhard Rapp. 2002. The computation of word associations: Comparing syntagmatic and paradigmatic approaches. In *Proc. COLING '02*, pages 1–7, Taipei, Taiwan.

Hinrich Schütze. 1993. Word space. In *Advances in Neural Information Processing Systems 5*, pages 895–902, Denver, Colorado, USA.

Benno Stein, Martin Potthast, and Martin Trenkmann. 2010. Retrieving Customary Web Language to Assist Writers. In *Advances in Information Retrieval, ECIR 10*, pages 631–635, Milton Keynes, UK.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188.