DKPro Similarity: An Open Source Framework for Text Similarity

Daniel Bär[†], **Torsten Zesch**[†‡], **and Iryna Gurevych**[†‡]
[†]Ubiquitous Knowledge Processing Lab (UKP-TUDA)
Department of Computer Science, Technische Universität Darmstadt
[‡]Ubiquitous Knowledge Processing Lab (UKP-DIPF)
German Institute for Educational Research and Educational Information
www.ukp.tu-darmstadt.de

Abstract

We present *DKPro Similarity*, an open source framework for text similarity. Our goal is to provide a comprehensive repository of text similarity measures which are implemented using standardized interfaces. DKPro Similarity comprises a wide variety of measures ranging from ones based on simple n-grams and common subsequences to high-dimensional vector comparisons and structural, stylistic, and phonetic measures. In order to promote the reproducibility of experimental results and to provide reliable, permanent experimental conditions for future studies, DKPro Similarity additionally comes with a set of full-featured experimental setups which can be run out-of-the-box and be used for future systems to built upon.

1 Introduction

Computing text similarity is key to several natural language processing applications such as automatic essay grading, paraphrase recognition, or plagiarism detection. However, only a few text similarity measures proposed in the literature are released publicly, and those then typically do not comply with any standardization. We are currently not aware of any designated text similarity framework which goes beyond simple lexical similarity or contains more than a small number of measures, even though related frameworks exist, which we discuss in Section 6. This fact was also realized by the organizers of the pilot *Semantic Textual Similarity Task* at SemEval-2012 (see Section 5), as they argue for the creation of an open source framework for text similarity (Agirre et al., 2012).

In order to fill this gap, we present DKPro Similarity, an open source framework for text similarity. DKPro Similarity is designed to complement DKPro Core[1], a collection of software components for natural language processing based on the Apache UIMA framework (Ferrucci and Lally, 2004). Our goal is to provide a comprehensive repository of text similarity measures which are implemented in a common framework using standardized interfaces. Besides the already available measures, DKPro Similarity is easily extensible and intended to allow for custom implementations, for which it offers various templates and examples. The Java implementation is publicly available at Google Code[2] under the Apache Software License v2 and partly under GNU GPL v3.

2 Architecture

DKPro Similarity is designed to operate in either of two modes: The *stand-alone mode* allows to use text similarity measures as independent components in any experimental setup, but does not offer means for further language processing, e.g. lemmatization. The *UIMA-coupled mode* tightly integrates similarity computation with full-fledged *Apache UIMA*-based language processing pipelines. That way, it allows to perform any number of languge processing steps, e.g. coreference or named-entitiy resolution, along with the text similarity computation.

Stand-alone Mode In this mode, text similarity measures can be used independently of any language processing pipeline just by passing them a pair of texts as (i) two strings, or (ii) two lists of strings (e.g. already lemmatized texts). We therefore provide an API module, which contains Java interfaces and abstract base classes for the measures. That way, DKPro Similarity allows for a maximum flexibility in experimental design, as the text similarity measures can easily be integrated with any existing experimental setup:

[1] code.google.com/p/dkpro-core-asl
[2] code.google.com/p/dkpro-similarity-asl

```
1 TextSimilarityMeasure m =
      new GreedyStringTiling();
2 double similarity =
      m.getSimilarity(text1, text2);
```

The above code snippet instantiates the *Greedy String Tiling* measure (Wise, 1996) and then computes the text similarity between the given pair of texts. The resulting similarity score is normalized into $[0, 1]$ where 0 means *not similar at all*, and 1 corresponds to *perfectly similar*.[3] By using the common `TextSimilarityMeasure` interface, it is easy to replace Greedy String Tiling with any measure of choice, such as *Latent Semantic Analysis* (Landauer et al., 1998) or *Explicit Semantic Analysis* (Gabrilovich and Markovitch, 2007). We give an overview of measures available in DKPro Similarity in Section 3.

UIMA-coupled Mode In this mode, DKPro Similarity allows text similarity computation to be directly integrated with any UIMA-based language processing pipeline. That way, it is easy to use text similarity components in addition to other UIMA-based components in the same pipeline. For example, an experimental setup may require to first compute text similarity scores and then to run a classification algorithm on the resulting scores.

In Figure 1, we show a graphical overview of the integration of text similarity measures (right) with a UIMA-based pipeline (left). The pipeline starts by reading a given dataset, then performs any number of pre-processing steps such as tokenization, sentence splitting, lemmatization, or stopword filtering, then runs the text similarity computation, before executing any subsequent post-processing steps and finally returning the processed texts in a suitable format for evaluation or manual inspection. As all text similarity measures in DKPro Similarity conform to standardized interfaces, they can be easily exchanged in the text similarity computation step.

With DKPro Similarity, we offer various subclasses of the generic UIMA components which are specifically tailored towards text similarity experiments, e.g. corpus readers for standard evaluation datasets as well as evaluation components for running typical evaluation metrics. By leveraging UIMA's architecture, we also define an

[3]Some string distance measures such as the *Levenshtein distance* (Levenshtein, 1966) return a raw distance score where less distance corresponds to higher similarity. However, the score can easily be normalized, e.g. by text length.

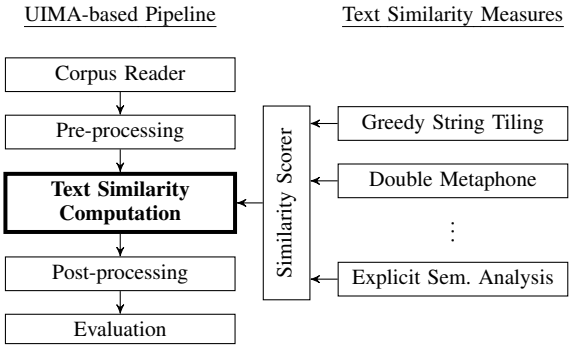

Figure 1: DKPro Similarity allows to integrate any text similarity measure (right) which conforms to standardized interfaces into a UIMA-based language processing pipeline (left) by means of a dedicated *Similarity Scorer* component (middle).

additional interface to text similarity measures: The `JCasTextSimilarityMeasure` inherits from `TextSimilarityMeasure`, and adds a method for two `JCas` text representations:[4]

```
double getSimilarity
    (JCas text1, JCas text2);
```

The additional interface allows to implement measures which have full access to UIMA's document structure. That way, it is possible to create text similarity measures which can use any piece of information that has been annotated in the processed documents, such as dependency trees or morphological information. We detail the new set of components offered by DKPro Similarity in Section 4.

3 Text Similarity Measures

In this section, we give an overview of the text similarity measures which are already available in DKPro Similarity. While we provide new implementations for a multitude of measures, we rely on specialized libraries such as the *S-Space Package* (see Section 6) if available. Due to space limitations and due to the fact that the framework is actively under development, we do not provide an exhaustive list here, but rather mention the most interesting and most popular measures.

3.1 Simple String-based Measures

DKPro Similarity includes text similarity measures which operate on string sequences and determine, for example, the longest common

[4]The `JCas` is an object-oriented Java interface to the *Common Analysis Structure* (Ferrucci and Lally, 2004), Apache UIMA's internal document representation format.

(non-)contiguous sequence of characters. It also contains *Greedy String Tiling* (Wise, 1996), a measure which allows to compare strings if parts have been reordered. The framework also offers measures which compute sets of character and word n-grams and compare them using different overlap coefficients, e.g. the Jaccard index. It further includes popular string distance metrics such as the *Jaro-Winkler* (Winkler, 1990), *Monge and Elkan (1997)* and *Levenshtein (1966)* distance measures.

3.2 Semantic Similarity Measures

DKPro Similarity also contains several measures which go beyond simple character sequences and compute text similarity on a semantic level.

Pairwise Word Similarity These measures are based on pairwise word similarity computations which are then aggregated for the complete texts. The measures typically operate on a graph-based representation of words and the semantic relations among them within a lexical-semantic resource. DKPro Similarity therefore contains adapters for WordNet, Wiktionary[5], and Wikipedia, while the framework can easily be extended to other data sources that conform to a common interface (Garoufi et al., 2008). Pairwise similarity measures in DKPro Similarity include *Jiang and Conrath (1997)* or *Resnik (1995)*. The aggregation for the complete texts can for example be done using the strategy by Mihalcea et al. (2006).

Vector Space Models These text similarity measures project texts onto high-dimensional vectors which are then compared. *Cosine similarity*, a basic measure often used in information retrieval, weights words according to their term frequencies or *tf-idf* scores, and computes the cosine between two text vectors. *Latent Semantic Analysis* (Landauer et al., 1998) alleviates the inherent sparseness of a high-dimensional term-document matrix by reducing it to one of reduced rank. *Explicit Semantic Analysis* (Gabrilovich and Markovitch, 2007) constructs the vector space on corpora where the documents are assumed to describe natural concepts such as *cat* or *dog*. Originally, Wikipedia was proposed as the document collection of choice.

DKPro Similarity goes beyond a single implementation of these measures and comes with highly customizable code which allows to set various parameters for the construction of the vector space and the comparison of the document vectors, and further allows to construct the vector space for arbitrary collections, e.g. domain-specific corpora.

[5] `http://www.wiktionary.org`

3.3 Further Measures

Previous research (Bär et al., 2012b) has shown promising results for the inclusion of measures which go beyond textual content and compute similarity along other text characteristics. Thus, DKPro Similarity also includes measures for structural, stylistic, and phonetic similarity.

Structural Similarity Structural similarity between texts can be computed, for example, by comparing sets of *stopword n-grams* (Stamatatos, 2011). The idea here is that similar texts may preserve syntactic similarity while exchanging only content words. Other measures in DKPro Similarity allow to compare texts by *part-of-speech n-grams*, and *order* and *distance features* for pairs of words (Hatzivassiloglou et al., 1999).

Stylistic Similarity DKPro Similarity includes, for example, a measure which compares *function word frequencies* (Dinu and Popescu, 2009) between two texts. The framework also includes a set of measures which capture statistical properties of texts such as the *type-token ratio* (TTR) and the *sequential TTR* (McCarthy and Jarvis, 2010).

Phonetic Similarity DKPro Similarity also allows to compute text similarity based on pairwise phonetic comparisons of words. It therefore contains implementations of well-known phonetic algorithms such as *Double Metaphone* (Philips, 2000) and *Soundex* (Knuth, 1973), which also conform to the common text similarity interface.

4 UIMA Components

In addition to a rich set of text similarity measures as partly described above, DKPro Similarity includes components which allow to integrate text similarity measures with any UIMA-based pipeline, as outlined in Figure 1. In the following, we introduce these components along with their resources.

Readers & Datasets DKPro Similarity includes corpus readers specifically tailored towards combining the input texts in a number of ways, e.g. all possible combinations, or each text paired with n others by random. Standard datasets for which

readers come pre-packaged include, among others, the SemEval-2012 STS data (Agirre et al., 2012), the METER corpus (Clough et al., 2002), or the RTE 1–5 data (Dagan et al., 2006). As far as license terms allow redistribution, the datasets themselves are integrated into the framework.

Similarity Scorer The *Similarity Scorer* allows to integrate any text similarity measure (which is decoupled from UIMA by default) into a UIMA-based pipeline. It builds upon the standardized text similarity interfaces and thus allows to easily exchange the text similarity measure as well as to specify the data types the measure should operate on, e.g. tokens or lemmas.

Machine Learning Previous research (Agirre et al., 2012) has shown that different text similarity measures can be combined using machine learning classifiers. Such a combination shows improvements over single measures due to the fact that different measures capture different text characteristics. DKPro Similarity thus provides adapters for the Weka framework (Hall et al., 2009) and allows to first pre-compute sets of text similarity scores which can then be used as features for various machine learning classifiers.

Evaluation Metrics In the final step of a UIMA pipeline, the processed data is read by a dedicated evaluation component. DKPro Similarity ships with a set of components which for example compute Pearson or Spearman correlation with human judgments, or apply task-specific metrics such as average precision as used in the RTE challenges.

5 Experimental Setups

DKPro Similarity further encourages the creation and publication of complete experimental setups. That way, we promote the reproducibility of experimental results, and provide reliable, permanent experimental conditions which can benefit future studies and help to stimulate the reuse of particular experimental steps and software modules.

The experimental setups are instantiations of the generic UIMA-based language processing pipeline depicted in Figure 1 and are designed to precisely match the particular task at hand. They thus come pre-configured with corpus readers for the relevant input data, with a set of pre- and post-processing as well as evaluation components, and with a set of text similarity measures which are well-suited for the particular task. The experimental setups are self-contained systems and can be run out-of-the-box without further configuration.[6]

DKPro Similarity contains two major types of experimental setups: (i) those for an *intrinsic evaluation* allow to evaluate the system performance in an isolated setting by comparing the system results with a human gold standard, and (ii) those for an *extrinsic evaluation* allow to evaluate the system with respect to a particular task at hand, where text similarity is a means for solving a concrete problem, e.g. recognizing textual entailment.

Intrinsic Evaluation DKPro Similarity contains the setup (Bär et al., 2012a) which participated in the *Semantic Textual Similarity (STS) Task* at SemEval-2012 (Agirre et al., 2012) and which has become one of the recommended baseline systems for the second task of this series.[7] The system combines a multitude of text similarity measures of varying complexity using a simple log-linear regression model. The provided setup allows to evaluate how well the system output resembles human similarity judgments on short texts which are taken from five different sources, e.g. paraphrases of news texts or video descriptions.

Extrinsic Evaluation Our framework includes two setups for an extrinsic evaluation: detecting text reuse, and recognizing textual entailment.

For detecting text reuse (Clough et al., 2002), the setup we provide (Bär et al., 2012b) combines a multitude of text similarity measures along different text characteristics. Thereby, it not only combines simple string-based and semantic similarity measures (see Sections 3.1 and 3.2), but makes extensive use of measures along structural and stylistic text characteristics (see Section 3.3). Across three standard evaluation datasets, the system consistently outperforms all previous work.

For recognizing textual entailment, we provide a setup which is similar in configuration to the one described above, but contains corpus readers and evaluation components precisely tailored towards the RTE challenge series (Dagan et al., 2006). We believe that our setup can be used for filtering those text pairs which need further analysis by a dedicated textual entailment system.

[6] A one-time setup of local lexical-semantic resources such as WordNet may be necessary, though.

[7] In 2013, the STS Task is a shared task of the Second Joint Conference on Lexical and Computational Semantics, `http://ixa2.si.ehu.es/sts`

6 Related Frameworks

To the best of our knowledge, only a few generalized similarity frameworks exist at all. In the following, we discuss them and give insights where DKPro Similarity uses implementations of these existing libraries. That way, DKPro Similarity brings together the scattered efforts by offering access to all measures through common interfaces. It goes far beyond the functionality of the original libraries as it generalizes the resources used, allows a tight integration with any UIMA-based pipeline, and comes with full-featured experimental setups which are pre-configured stand-alone text similarity systems that can be run out-of-the-box.

S-Space Package Even though no designated text similarity library, the S-Space Package (Jurgens and Stevens, 2010)[8] contains some text similarity measures such as Latent Semantic Analysis (LSA) and Explicit Semantic Analysis (see Section 3.2). However, it is primarily focused on word space models which operate on word distributions in text. Besides such algorithms, it offers a variety of interfaces, data structures, evaluation datasets and metrics, and global operation utilities e.g. for dimension reduction using Singular Value Decomposition or randomized projections, which are particularly useful with such distributional word space models. DKPro Similarity integrates LSA based on the S-Space Package.

Semantic Vectors The Semantic Vectors package is a package for distributional semantics (Widdows and Cohen, 2010)[9] that contains measures such as LSA and allows for comparing documents within a given vector space. The main focus lies on word space models with a number of dimension reduction techniques, and applications on word spaces such as automatic thesaurus generation.

WordNet::Similarity The open source package by Pedersen et al. (2004)[10] is a popular Perl library for the similarity computation on WordNet. It comprises six word similarity measures that operate on WordNet, e.g. *Jiang and Conrath (1997)* or *Resnik (1995)*. Unfortunately, no strategies have been added to the package yet which aggregate the word similarity scores for complete texts in a similar manner as described in Section 3.2. In DKPro Similarity, we offer native Java implementations of all measures contained in WordNet::Similarity, and allow to go beyond WordNet and use the measures with any lexical-semantic resource of choice, e.g. Wiktionary or Wikipedia.

SimMetrics Library The Java library by Chapman et al. (2005)[11] exclusively comprises text similarity measures which compute lexical similarity on string sequences and compare texts without any semantic processing. It contains measures such as the *Levenshtein (1966)* or *Monge and Elkan (1997)* distance metrics. In DKPro Similarity, some string-based measures (see Section 3.1) are based on implementations from this library.

SecondString Toolkit The freely available library by Cohen et al. (2003)[12] is similar to SimMetrics, and also implemented in Java. It also contains several well-known text similarity measures on string sequences, and includes many of the measures which are also part of the SimMetrics Library. Some string-based measures in DKPro Similarity are based on the SecondString Toolkit.

7 Conclusions

We presented DKPro Similarity, an open source framework designed to streamline the development of text similarity measures. All measures conform to standardized interfaces and can either be used as stand-alone components in any experimental setup (e.g. an already existing system which is not based on Apache UIMA), or can be tightly coupled with a full-featured UIMA-based language processing pipeline in order to allow for advanced processing capabilities.

We would like to encourage other researchers to participate in our efforts and invite them to explore our existing experimental setups as outlined in Section 5, run modified versions of our setups, and contribute own text similarity measures to the framework. For that, DKPro Similarity also comes with an example module for getting started, which guides first-time users through both the stand-alone and the UIMA-coupled modes.

Acknowledgements This work has been supported by the Volkswagen Foundation as part of the Lichtenberg Professorship Program under grant No. I/82806, and by the Klaus Tschira Foundation under project No. 00.133.2008. We thank Richard Eckart de Castilho and all other contributors.

[8] code.google.com/p/airhead-research
[9] code.google.com/p/semanticvectors
[10] sourceforge.net/projects/wn-similarity
[11] sourceforge.net/projects/simmetrics
[12] sourceforge.net/projects/secondstring

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 Task 6: A Pilot on Semantic Textual Similarity. In *Proc. of the 6th Int'l Works. on Semantic Eval.*, pages 385–393.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012a. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. In *Proc. of the 6th Int'l Workshop on Semantic Evaluation*, pages 435–440.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. 2012b. Text Reuse Detection Using a Composition of Text Similarity Measures. In *Proc. of the 24th Int'l Conf. on Computational Linguistics*, pages 167–184.

Sam Chapman, Barry Norton, and Fabio Ciravegna. 2005. Armadillo: Integrating Knowledge for the Semantic Web. In *Proceedings of the Dagstuhl Seminar in Machine Learning for the Semantic Web*.

Paul Clough, Robert Gaizauskas, Scott S.L. Piao, and Yorick Wilks. 2002. METER: MEasuring TExt Reuse. In *Proceedings of ACL*, pages 152–159.

William W. Cohen, Pradeep Ravikumar, and Stephen Fienberg. 2003. A Comparison of String Metrics for Matching Names and Records. In *Proc. of KDD Works. on Data Cleaning and Object Consolidation*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges*, Lecture Notes in Computer Science, pages 177–190.

Liviu P. Dinu and Marius Popescu. 2009. Ordinal measures in authorship identification. In *Proceedings of the 3rd PAN Workshop. Uncovering Plagiarism, Authorship and Social Software Misuse*, pages 62–66.

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.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of IJCAI*, pages 1606–1611, Hyderabad, India.

Konstantina Garoufi, Torsten Zesch, and Iryna Gurevych. 2008. Representational Interoperability of Linguistic and Collaborative Knowledge Bases. In *Proceedings of the KONVENS Workshop on Lexical-Semantic and Ontological Resources*.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1):10–18.

Vasileios Hatzivassiloglou, Judith L. Klavans, and Eleazar Eskin. 1999. Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning. In *Proceedings of EMNLP/VLC*, pages 203–212.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of ROCLING*, pages 19–33.

David Jurgens and Keith Stevens. 2010. The S-Space Package: An Open Source Package for Word Space Models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 30–35, Uppsala, Sweden.

Donald E. Knuth. 1973. *The Art of Computer Programming: Volume 3, Sorting and Searching*. Addison-Wesley.

Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. 1998. An Introduction to Latent Semantic Analysis. *Discourse Processes*, 25(2):259–284.

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Philip M. McCarthy and Scott Jarvis. 2010. MTLD, vocd-D, and HD-D: A validation study of sophisticated approaches to lexical diversity assessment. *Behavior research methods*, 42(2):381–392.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In *Proceedings of AAAI-06*, pages 775–780, Boston, MA, USA.

Alvaro Monge and Charles Elkan. 1997. An efficient domain-independent algorithm for detecting approximately duplicate database records. In *Proceedings of the SIGMOD Workshop on Data Mining and Knowledge Discovery*, pages 23–29.

Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. WordNet::Similarity - Measuring the Relatedness of Concepts. In *Proceedings of the HLT-NAACL: Demonstration Papers*, pages 38–41.

Lawrence Philips. 2000. The double metaphone search algorithm. *C/C++ Users Jour.*, 18(6):38–43.

Philip Resnik. 1995. Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In *Proceedings of the IJCAI*, pages 448–453.

Efstathios Stamatatos. 2011. Plagiarism detection using stopword n-grams. *Journal of the American Society for Information Science and Technology*, 62(12):2512–2527.

Dominic Widdows and Trevor Cohen. 2010. The Semantic Vectors Package: New Algorithms and Public Tools for Distributional Semantics. In *Proceedings of IEEE-ICSC*, pages 9–15.

William E. Winkler. 1990. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. In *Proceedings of the Survey Research Methods Section*, pages 354–359.

Michael J. Wise. 1996. YAP3: Improved detection of similarities in computer program and other texts. In *Proc. of the 27th SIGCSE Technical Symposium on Computer Science Education*, pages 130–134.