DARIAH-DKPro-Wrapper Output Format (DOF) Specification

Fotis Jannidis, Stefan Pernes, Steffen Pielström, Isabella Reger, Nils Reimers, Thorsten Vitt

DARIAH-DE Working Papers Nr. 20

Jannidis, F., Pernes, S., Pielström, S., Reger, I., Reimers, N., Vitt, T.: DARIAH-DKPro-Wrapper Output Format (DOF) Specification
Göttingen : GOEDOC, Dokumenten- und Publikationsserver der Georg-August-Universität, 2016
(DARIAH-DE working papers 20)

Verfügbar:
PURL: http://resolver.sub.uni-goettingen.de/purl/?dariah-2016-6
URN: http://nbn-resolving.de/urn:nbn:de:gbv:7-dariah-2016-6-2

Bibliographische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliographie; detaillierte bibliographische Daten sind im Internet über <http://dnb.ddb.de> abrufbar.

Erschienen in der Reihe
DARIAH-DE working papers

ISSN: 2198-4670

Herausgeber der Reihe
DARIAH-DE, Niedersächsische Staats- und Universitätsbibliothek

Mirjam Blümm, Thomas Kollatz, Stefan Schmunk und Christof Schöch

---

Abstract: The DARIAH-DKPro-Wrapper Output Format (DOF) is a tab-separated file format, designed to be easily accessible by various analysis tools and scripting languages. It is based on a modular linguistic processing pipeline, which includes a range of analysis capabilities and has been fitted to book-length documents. Both processing pipeline and output format have been developed in cooperation by the Department of Literary Computing, Wü rzburg ("Lehrstuhl Für Computerphilologie Und Neuere Deutsche Literaturgeschichte. Universität Würzburg" 2016) and the Ubiquitous Knowledge Processing Lab, Darmstadt ("Ubiquitous Knowledge Processing Lab. Technische Universität Darmstadt" 2016) as part of DARIAH-DE, Digital Research Infrastructure for the Arts and Humanities ("DARIAH-DE" 2016).

Keywords: NLP Pipeline, Benutzerfreundlichkeit, DKPro

NLP pipeline, user friendliness, DKPro

DARIAH-DKPro-Wrapper Output Format (DOF) Specification

Fotis Jannidis[1] Stefan Pernes[1] Steffen Pielström[1]

Isabella Reger[1] Nils Reimers[2] Thorsten Vitt[1]

[1]Lehrstuhl für Computerphilologie und Neuere Deutsche Literaturgeschichte, Universität Würzburg
[2]Ubiquitous Knowledge Processing Lab, Technische Universität Darmstadt

Fotis Jannidis, Stefan Pernes, Steffen Pielström, Isabella Reger, Nils Reimers, Thorsten Vitt: "DARIAH-DKPro-Wrapper Output Format (DOF) Specification". *DARIAH-DE Working Papers* No. 20. Göttingen: DARIAH-DE, 2016. URN: urn:nbn:de:gbv:7-dariah-2016-6-2.

DARIAH-DKPro-Wrapper and the DARIAH-DKPro-Wrapper Output Format have been developed within the DARIAH-DE ("DARIAH-DE" 2016) initiative, the German contribution to DARIAH-EU ("DARIAH-EU" 2016), European Digital Research Infrastructure for the Arts and Humanities. Funding has been provided by the German Federal Ministry for Research and Education (BMBF) under the identifier 01UG1110A-N.

Abstract

The DARIAH-DKPro-Wrapper Output Format (DOF) is a tab-separated file format, designed to be easily accessible by various analysis tools and scripting languages. It is based on a modular linguistic processing pipeline, which includes a range of analysis capabilities and has been fitted to book-length documents. Both processing pipeline and output format have been developed in cooperation by the Department of Literary Computing, Würzburg ("Lehrstuhl Für Computerphilologie Und Neuere Deutsche Literaturgeschichte. Universität Würzburg" 2016) and the Ubiquitous Knowledge Processing Lab, Darmstadt ("Ubiquitous Knowledge Processing Lab. Technische Universität Darmstadt" 2016) as part of DARIAH-DE, Digital Research Infrastructure for the Arts and Humanities ("DARIAH-DE" 2016).

Schlagwörter

NLP Pipeline, Benutzerfreundlichkeit, DKPro

Keywords

NLP pipeline, user friendliness, DKPro

Contents

1 License

The DOF standard is licensed under the Apache License, Version 2.0 ("Apache License Version 2.0," n.d.).

2 About DARIAH-DKPro-Wrapper

With the increasing availability of digitized literary texts, computational texts analysis more and more becomes a methodological option in literary studies. Thus, the ability to use script languages such as Python ("The Python Language" 2016) or R ("The R Project" 2016) for quantitative text analysis becomes increasingly widespread among digital literary scholars. Many analytical methods, however, require linguistic information. Various natural language processing (NLP) tools allow to automatically extract different kinds of linguistic information – for example, parts-of-speech, lemmatised forms, and named entities can be automatically derived from a text. However, one would need to string those individual tools together. The DARIAH-DKPro-Wrapper (DDW) combines them into a single command-line program computing word-by-word linguistic annotation, thereby considerably simplifying the generation of such annotations. We present the output format of the DDW. It is designed as a versatile, easy-to-use link between NLP preprocessing and working in scripting languages like Python or R.

The DDW is based on Apache UIMA ("Apache UIMA" 2016) and DKPro Core ("Darmstadt Knowledge Processing Repository" 2016). UIMA is a framework for the management of unstructured information like natural language text. UIMA enables applications to be decomposed into components, for example a component to read a document, components for different linguistic annotations and a component to store the results in a desired format. All components can be plugged together and can then be executed as a pipeline. DKPro Core is based on UIMA and integrates several well known NLP tools, including Stanford CoreNLP (Manning et al. 2014), OpenNLP ("Apache OpenNLP" 2016), and Mate Tools (Bohnet and Nivre 2012). DKPro Core makes it easy for the programmer to choose between different NLP tools and to use the output of different NLP tools for further processing. DDW makes it easy for the user: it allows to read in documents from arbitrary text files, which are then processed in the UIMA pipeline and enriched with linguistic information. For further analysis, the result is written to a tab-separated file format, where each row corresponds to one token in the document and each column corresponds to one layer of analysis (cf. the example data in the appendix). By default, all configurable components in the DDW are enabled. They consist of: segmentation, part-of-speech tagging, lemmatisation, chunking, morphology tagging, hyphenation, named entity recognition, dependency parsing, consituency parsing, semantic role labeling, and coreference resolution. However, the program can be customised using configuration files, which are predefined and readily available for a number of languages. Using these files, components can be swapped, configured – for example to load custom models – and turned off, to deactivate expensive processing steps when they are not needed. As of DKPro Core 1.8.0, more than 40 components and support for over 30 languages is available. For a detailed list of components and the languages they support, see ("DKPro Core Component Reference" 2016) and ("DKPro Core Model Reference" 2016). The DDW source code, alongside a compact user guide and a comprehensive tutorial

on various use cases, can be found at the official DDW Git Repository ("DARIAH-DE DKPro Wrapper" 2016).

3 Aims of the Output Format

The overall aim of DOF is to provide a file format that is as easy to process as possible, while offering the full range of linguistic information provided by the analysis components used. It has been developed with the following goals in mind:

- To be suitable for the representation of various linguistic annotations and data types;
- To provide auxiliaries for the processing of book-length documents;
- To be human-readable and self-documenting;
- To be easily processable by widely-used scripting languages and data science tools;
- To have as little computational overhead as possible;
- To be extensible.

4 General Characteristics

DOF is based on the CoNLL format (in its 2009 version ("CoNLL 2009 Shared Task Description" 2009)), which is a frequently used format in NLP. Its line-based structure – where each line represents a token – and the concept of annotation layers as columns, stays close to the original CoNLL layout. It has the advantage of leveraging various analysis tools already built around this kind of structure. Additionally, DOF is constructed as a TSV (tab-separated values) ("Definition of Tab-Separated-Values (TSV)," n.d.) file, which increases possibilities for ingestion – one can choose between numerous scripting languages, analysis packages, and it can even be read by standard spreadsheet applications. In contrast to the CoNLL layout, there is no empty line after each sentence. Such empty lines are difficult to process in most existent TSV readers. Instead, each row includes an ascending ID of the sentence.

Primarily, the DDW and its output format were devised for the efficient processing of book-length documents. Thus, various segmentation strategies are built-in, in order to facilitate the traversal of such large text files and to provide means for automatic selection of specific segments, e.g. programmatically by use of GroupBy methods.

- Sentences are counted in the SentenceId column in order to allow for a continuous placement of rows – in contrast to the CoNLL 2009 layout, which introduces empty lines after each sentence.
- In case the input file(s) processed by DDW are XML files, the SectionId field contains the XPath ("The XML Path Language" 2015) for each token.

- In addition to segmentation using sentence identifiers, a pattern-matching based segmentation strategy is introduced: The ParagraphId column holds an identifier describing which paragraph each token belongs to. It can be set to count either single or double line breaks in the original document.
- Another pattern-based feature is the quotation detection which uses the QuoteMarker column to indicate for each token whether it is inside or outside a quotation.
- In order to allow for the lookup of specific tokens, the format includes a TokenId, as well as an offset in the original document corresponding to the token in question, which is indicated in the Begin and End columns.
- Also, the format is able to hold grammatical information represented as trees for each sentence. Not only "flat" dependency trees (DependencyHead, DependencyRelation), but also nested phrase structures generated by a constituency parser. The corresponding SyntaxTree column resembles the "parse bit" of CoNLL 2012 ("CoNLL 2012 Shared Task Data" 2012). Example code for compiling graph objects out of the information provided by dependency and constituency parsers can be found in the DDW's documentation ("DARIAH-DE DKPro Wrapper Tutorial: NLP Based Analysis of Literary Texts" 2015).

5 Relation to other Approaches

During the development of the DDW, various approaches were discussed and tested before the file format attained its current form. A key problem is bridging the gap between the low-level flexibility of the CAS ("Apache UIMA CAS Interface" 2010) structure (which is the internal format UIMA frameworks such as DKPro are based on) and a format that is straightforward to process using a variety of analysis tools, while retaining all of the information contained in the CAS structure.

In principle, there are a number of possible alternatives to the chosen TSV file format. Tree-based (e.g. GrAF, Ide and Suderman 2007) and XML-based (e.g. TIGER/SALSA XML, Erk and Pado 2004) formats allow for a more native representation of tree structures and nested annotations, while a modern binary format such as docrep (Dawborn and Curran 2014) even adds an improved processing speed to that. Nevertheless, those formats are only really accessible programmatically, can be computationally expensive and are not necessarily human-readable, all of which make them more cumbersome to process for the user primarily interested in data analysis. The proposed DOF format is in many respects similar to the output format of BookNLP ("BookNLP: Natural Language Processing Pipeline for Book-Length Documents" 2014), another natural language processing pipeline for book-length documents, which, however, is based on components specific to the English language.

6 Format Specification

DOF files are UTF-8 encoded text files in which each row represents a single token. The first row contains column headers in form of the field names according to the following table. The columns are separated

by one tab (`\t`) character. Column values never contain Tab characters, so there is no quoting and no escaping. Columns can be addressed by their index number, which can vary depending on the number of components the pipeline is configured for, as well as by the columns' field names.

The full column layout is shown below:

Field Name	Description
SectionId	For XML input files, this column contains the XPath of the token. For text files, this column is empty.
ParagraphId	Ascending number for the paragraphs in the document. Starting at 0.
SentenceId	Ascending number for the sentences in the document. Starting at 0.
TokenId	Ascending number for the tokens in the document. Starting at 0.
Begin	Offset in characters for the begin of the token.
End	Offset in characters for the end of the token.
Token	The token itself.
Lemma	The lemma of the token.
CPOS	Coarse grained part of speech information. Tagset reproduced in the Appendix.
POS	Part of speech information as assigned by the POS tagger. Content varies according to the tagset used by selected POS tagger component. See the list of models available in DKPro ("DKPro Core Model Reference" 2016).
Chunk	Chunking information, BIO (Begin/In/Out) ("CoNLL 2000 Shared Task Description" 2000) encoded. Thus, B_CHUNK marks the first word of the chunk, I_CHUNK marks each other word in the chunk, and O_CHUNK marks words outside the chunk.
Morphology	The Mate Tools Morphology Tagger (Bohnet et al. 2013) is used to derive morphological information for the token.
Hyphenation	The LaTeX hyphenation algorithm is used to detect the syllables of the token. The syllables of the token are separated via hyphens.
DependencyHead	Head (token id) of the current token.
DependencyRelation	Dependency relation to the head. Content varies according to the tagset used by selected dependency parser component. See the list of models available in DKPro ("DKPro Core Model Reference" 2016).
NamedEntity	Named Entity information, BIO encoded.
QuoteMarker	1 if token is inside quotes, 0 else.

Field Name	Description
CoreferenceChainIds	If the token is part of a coreference chain, the according Id of the chain is noted in this field. The field uses BIO encoding and may contain multiple chain ids. Chain ids are separated by comma.
SyntaxTree	Bracketed structure broken before the first open parenthesis in the parse, and the word/part_of_speech leaf is replaced with a *. Identical format as for the CoNLL 2012 parse bit.
Predicate	Predicate information for the current token. Consists of the verb and its mapping to a FrameNet ("The FrameNet Project" 2016) sense.
SemanticArgument Index	Index of the semantic arguments for this predicate (0 ... N)
SemanticArgument0 ... N	The labels for the semantic arguments for the detected predicates are written in individual columns. The argument for the n-th predicate is written to the n-th column for semantic arguments. The labels follow the PropBank_Style (A0, A1,...).

7 Conclusion

The DDW provides a simple all-in-one tool for many popular linguistic analysis steps. Its output format provides an easy starting point for further analyses, aligning the output of the various tools in a format that can be processed using standard libraries or even spreadsheet software with as little computational overhead as possible, while also being human-readable. Extensive documentation included with the DDW illustrates some practical applications of the wrapper and the format, showing how to read DOF files in R and Python and how to interpret the various fields.

8 Appendix

8.1 Coarse grained Part-of-Speech Tagset

All individual tagsets output by different components map onto a restricted, universal part-of-speech tagset ("DKPro Core Part-of-Speech Tagset" 2016):

- *ADJ* - Adjective
- *ADV* - Adverb
- *ART* - Article
- *CARD* - Numeral

- *CONJ* - Conjunction
- *N* - Noun
- *NN* - Common noun
- *NP* - Noun phrase
- *O* - Catch-all for other categories such as abbreviations or foreign words
- *PP* - Prepositions and postpositions
- *PR* - Pronoun
- *PRT* - Particle
- *PUNC* - Punctuation
- *V* - Verb

Example Output German: Theodor Fontane - Effi Briest (1894/1895)

SectionId	ParagraphId	SentenceId	TokenId	Begin	End	Token	Lemma	CPOS	POS	Chunk	Morphology	Hyphenation	DependencyHead	DependencyRelation	NamedEntity	QuoteMarker	CoreferenceChainIds	SyntaxTree	Predicate	SemanticArgumentIndex
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	156	1259	1261	In	in	PP	APPR	_	_	In	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	157	1262	1267	Front	Front	NN	NN	_	sg|3|pres|ind	Front	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	158	1268	1271	des	der	ART	ART	_	gen|sg|masc	des	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	159	1272	1277	schon	schon	ADV	ADV	_	_	schon	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	160	1278	1282	seit	seit	PP	APPR	_	_	seit	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	161	1283	1291	Kurf,rst	Kurf,rst	NN	NN	_	nom|sg|masc	Kur-f,rst	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	162	1292	1297	Georg	Georg	NP	NE	_	nom|sg|masc	Ge-org	_	_	B-PER	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	163	1298	1305	Wilhelm	Wilhelm	NP	NE	_	nom|sg|masc	Wil-helm	_	_	I-PER	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	164	1306	1309	von	von	PP	APPR	_	_	von	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	165	1310	1313	der	der	ART	ART	_	dat|sg|fem	der	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	166	1314	1321	Familie	Familie	NN	NN	_	dat|sg|fem	Fa-mi-lie	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	167	1322	1325	von	von	PP	APPR	_	_	von	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	168	1326	1332	Briest	Briest	NN	NN	_	pos	Briest	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	169	1333	1342	bewohnten	bewohnt	ADJ	ADJA	_	gen|sg|neut|pos	be-wohn-ten	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	170	1343	1355	Herrenhauses	Herrenhaus	NN	NN	_	gen|sg|neut	Her-ren-hau-ses	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	171	1356	1358	zu	zu	PP	APPR	_	_	zu	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	172	1359	1372	Hohen-Cremmen	Hohen-Cremmen	NN	NN	_	_	Ho-hen-?Crem-men	_	_	B-LOC	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	173	1373	1377	fiel	fallen	V	VVFIN	_	sg|3|past|ind	fiel	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	174	1378	1384	heller	heller	NP	NE	_	nom|sg|*	hel-ler	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	175	1385	1397	Sonnenschein	Sonnenschein	NP	NE	_	acc|sg|masc	Son-nen-schein	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	176	1398	1401	auf	auf	PP	APPR	_	_	auf	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	177	1402	1405	die	der	ART	ART	_	acc|sg|fem	die	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	178	1406	1419	mittagsstille	mittagsstill	ADJ	ADJA	_	acc|sg|fem|pos	mit-tags-stil-le	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	179	1420	1430	Dorfstrafle	Dorfstrafle	NN	NN	_	acc|sg|fem	Dorf-stra-fle	_	_	_	0	_	_	_	_
//TEI[1]/text[1]/body[1]/div[1]/div[2]/div[1]/p[1]	19	22	180	1430	1431			--	PUNC	$		_	_				_	_	_	0 _

Example Output English Florence L. Barclay - Through the postern gate; a romance in seven days (1912)

SectionId	ParagraphId	SentenceId	TokenId	Begin	End	Token	Lemma	CPOS	POS	Chunk	Morphology	Hyphenation	DependencyHead	DependencyRelation	NamedEntity	QuoteMarker	CoreferenceChainIds	SyntaxTree	Predicate	SemanticArgumentIndex	SemanticArgument0	SemanticArgument1
_	1	2	20	75	77	He	he	PR	PRP	B-NP	_	He	21	SBJ	_	0	_	_	_	_	A0	_
_	1	2	21	78	81	lay	lay	V	VBD	B-VP	_	lay	_	_	_	0	_	_	lay.01	_	0 _	_
_	1	2	22	82	86	back	back	PP	RP	B-ADVP	_	back	21	ADV	_	0	_	_	_	_	_	_
_	1	2	23	86	87	,	,	PUNC	,	_	_	,	21	P	_	0	_	_	_	_	_	_
_	1	2	24	88	90	in	in	PP	IN	B-PP	_	in	21	LOC	_	0	_	_	_	_	AM-LOC	_
_	1	2	25	91	92	a	a	ART	DT	B-NP	_	a	28	NMOD	_	0	_	_	_	_	_	_
_	1	2	26	93	97	deep	deep	ADJ	JJ	I-NP	_	deep	28	NMOD	_	0	_	_	_	_	_	_
_	1	2	27	98	104	wicker	wicker	NN	NN	I-NP	_	wick-er	28	NMOD	_	0	_	_	_	_	_	_
_	1	2	28	105	110	chair	chair	NN	NN	I-NP	_	chair	24	PMOD	_	0	_	_	_	_	_	_
_	1	2	29	110	111	,	,	PUNC	,	_	_	,	21	P	_	0	_	_	_	_	_	_
_	1	2	30	112	117	under	under	PP	IN	B-PP	_	un-der	21	LOC	_	0	_	_	_	_	AM-LOC	_
_	1	2	31	118	121	the	the	ART	DT	B-NP	_	the	33	NMOD	_	0	_	_	_	_	_	_
_	1	2	32	122	125	old	old	ADJ	JJ	I-NP	_	old	33	NMOD	_	0	_	_	_	_	_	_
_	1	2	33	126	139	mulberry-tree	mulberry-tree	NN	NN	I-NP	_	mul-ber-ry-â€‹tree	30	PMOD	_	0	_	_	_	_	_	_
_	1	2	34	139	140	.	.	PUNC	.	_	_	.	21	P	_	0	_	_	_	_	_	_

Bibliography

"Apache License Version 2.0." n.d. http://www.apache.org/licenses/LICENSE-2.0.

"Apache OpenNLP." 2016. http://opennlp.apache.org.

"Apache UIMA." 2016. http://uima.apache.org.

"Apache UIMA CAS Interface." 2010. https://uima.apache.org/downloads/releaseDocs/2.3.0-incubating/docs/api/org/apache/uima/cas/CAS.html.

Bohnet, Bernd, and Joakim Nivre. 2012. "A Transition-Based System for Joint Part-of-Speech Tagging and Labeled Non-Projective Dependency Parsing." In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 1455–65. Association for Computational Linguistics.

Bohnet, Bernd, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajič. 2013. "Joint Morphological and Syntactic Analysis for Richly Inflected Languages." *Transactions of the Association for Computational Linguistics* 1: 415–28.

"BookNLP: Natural Language Processing Pipeline for Book-Length Documents." 2014. http://github.com/dbamman/book-nlp.

"CoNLL 2000 Shared Task Description." 2000. http://www.cnts.ua.ac.be/conll2000/chunking/.

"CoNLL 2009 Shared Task Description." 2009. http://ufal.mff.cuni.cz/conll2009st/taskdescription.html.

"CoNLL 2012 Shared Task Data." 2012. http://conll.cemantix.org/2012/data.html.

"DARIAH-DE." 2016. http://de.dariah.eu.

"DARIAH-DE DKPro Wrapper." 2016. http://github.com/DARIAH-DE/DARIAHDKProWrapper.

"DARIAH-DE DKPro Wrapper Tutorial: NLP Based Analysis of Literary Texts." 2015. http://dariah-de.github.io/DARIAH-DKPro-Wrapper/tutorial.html.

"DARIAH-EU." 2016. http://dariah.eu.

"Darmstadt Knowledge Processing Repository." 2016. http://www.dkpro.org.

Dawborn, Tim, and James R Curran. 2014. "Docrep: A Lightweight and Efficient Document Representation Framework." In *COLING*, 762–71.

"Definition of Tab-Separated-Values (TSV)." n.d. https://www.iana.org/assignments/media-types/text/tab-separated-values.

"DKPro Core Component Reference." 2016. http://dkpro.github.io/dkpro-core/releases/1.8.0/docs/component-reference.html.

"DKPro Core Model Reference." 2016. http://dkpro.github.io/dkpro-core/releases/1.8.0/docs/model-reference.html.

"DKPro Core Part-of-Speech Tagset." 2016. http://dkpro.github.io/dkpro-core/releases/1.8.0/docs/

typesystem-reference.html#type-de.tudarmstadt.ukp.dkpro.core.api.lexmorph.type.pos.POS.

Erk, Katrin, and Sebastian Pado. 2004. "A Powerful and Versatile XML Format for Representing Role-Semantic Annotation." In *LREC*. Citeseer.

Ide, Nancy, and Keith Suderman. 2007. "GrAF: A Graph-Based Format for Linguistic Annotations." In *Proceedings of the Linguistic Annotation Workshop*, 1–8. Association for Computational Linguistics.

"Lehrstuhl Für Computerphilologie Und Neuere Deutsche Literaturgeschichte. Universität Würzburg." 2016. http://www.germanistik.uniwuerzburg.de/lehrstuehle/computerphilologie.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. "The Stanford CoreNLP Natural Language Processing Toolkit." In *Association for Computational Linguistics (ACL) System Demonstrations*, 55–60. http://www.aclweb.org/anthology/P/P14/P14-5010.

"The FrameNet Project." 2016. http://framenet.icsi.berkeley.edu.

"The Python Language." 2016. http://www.python.org.

"The R Project." 2016. http://www.rproject.org.

"The XML Path Language." 2015. http://www.w3.org/TR/xpath.

"Ubiquitous Knowledge Processing Lab. Technische Universität Darmstadt." 2016. http://www.ukp.tudarmstadt.de.