Web services and data mining: combining linguistic tools for Polish with an analytical platform

Maciej Ogrodniczuk
Institute of Computer Science, Polish Academy of Sciences
maciej.ogrodniczuk@ipipan.waw.pl

Abstract

In this paper we present a new combination of existing language tools for Polish with a popular data mining platform intended to help researchers from digital humanities perform computational analyses without any programming. The toolset includes RapidMiner Studio, a software solution offering graphical setup of integrated analytical processes and Multiservice, a Web service offering access to several state-of-the-art linguistic tools for Polish. The setting is verified in a simple task of counting frequencies of unknown words in a small corpus.

1 Introduction

Applying language technologies to data used by the humanities is not always easy for researchers with no technical background. Even though some are probably used to querying language corpora, using regular expressions or analysing tabular data, obstacles related to complex local installation and configuration of language tools are sometimes insurmountable. This is where Web services and Web applications come in handy but still, they offer fragmented solutions, often making result sets difficult to export or further analyse which in turn keeps most parts of the process manual.

In this paper we demonstrate how a robust linguistic Web service framework for Polish could be linked to a mature data science platform to offer both easy-to-start graphical user interface to test research hypotheses and a powerful analytical environment for data mining. Section 2 describes the tools being combined, Section 3 presents the motivation behind this step, Section 4 lists the configuration details and Section 5 offers a sample scenario illustrating the capabilities of the platform and showing how easily the setting can be used in daily tasks of a corpus linguist.

2 The toolset

2.1 RapidMiner

RapidMiner, formerly known as YALE (Yet Another Learning Environment), is a data mining platform developed at the Technical University of Dortmund (Mierswa et al., 2006) and successfully transformed into commercial application, currently one of the leaders in the Gartner Magic Quadrant for Advanced Analytics (Kart et al., 2016). Basic Edition of the platform (limited to 10,000 rows and 1 logical processor) is freely available as AGPL; cost-free licences can also be granted for educational purposes.

The main feature of the platform is its user-friendly interface (see Figure 1) which facilitates setting up complex processes by dragging-and-dropping configurable building blocks — *operators*, selected from the 1500 currently available. Operators offer procedures for data loading, text transformation, processing and visualisation, statistical analysis and many other sophisticated tasks. Data in RapidMiner are modelled as *examples* carrying certain *attributes* which corresponds to tabular representation with examples as rows and attributes as columns. Each processor requires certain data inputs and produces a number of outputs which makes chaining easy. New operators, free to use or available for a fee, can be integrated as extensions and are installed via RapidMiner Marketplace.

Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH),
pages 187–195, Osaka, Japan, December 11-17 2016.

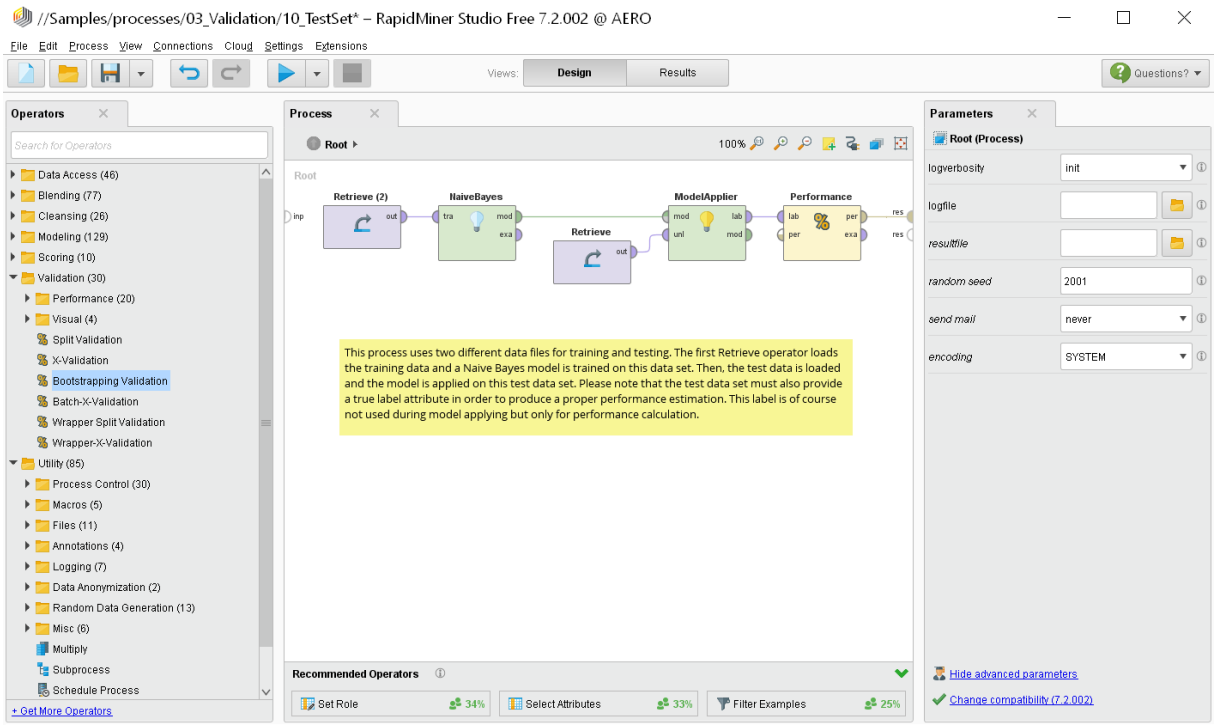

Figure 1: Graphical user interface of RapidMiner Studio

The usage of the environment is two-phased: in the first (design) phase the process is constructed graphically, with operators being ordered, configured with individual parameters and linked with input–output pipes. In the second phase the process is run and results displayed or saved to a file. Configuration of processes is saved in an XML file so it can be exported and moved between RapidMiner configurations. The workspace is intuitive and resembles programmer's desktop without the coding pane but keeping its capabilities such as setting breakpoints, browsing partial results etc.

2.2 Multiservice

Multiservice (Ogrodniczuk and Lenart, 2012) is a platform created in CLARIN to make offline language processing tools for Polish available as Web services and offer their chaining thanks to a common linguistic representation format and asynchronous execution architecture. As of 2016, the toolset comprises several disambiguating taggers (with paragraph-, sentence- and token-level segmentation and morphological analysis): Pantera (Acedański, 2010), WMBT (Radziszewski and Śniatowski, 2011), Concraft (Waszczuk, 2012), WCRFT (Radziszewski, 2013), ensemble tagger PoliTa (Kobyliński, 2014), sentiment analyser Sentipejd (Buczyński and Wawer, 2008), dependency parser (Wróblewska, 2014), shallow parser Spejd (Przepiórkowski and Buczyński, 2007), named entity recognizer Nerf (Waszczuk et al., 2013), two coreference resolvers Ruler (Ogrodniczuk and Kopeć, 2011) and Bartek (Ogrodniczuk et al., 2015, chapter 12), OpenTextSummarizer (Rotem, 2003) adjusted for Polish and two other summarization tools: Lakon (Dudczak, 2007), Świetlicka's summarizer (Świetlicka, 2010) and Nicolas (Kopeć, 2016).

Interaction with Multiservice is currently available via a dedicated API available in Java and Python or a Web demo of the service (`http://multiservice.nlp.ipipan.waw.pl/`). Both these methods have their drawbacks: programmatical access is in most cases too difficult to use by representatives of the humanities and the Web application offers only a brat-based (Stenetorp et al., 2012) interface showing layers of annotations in separate tabs plus a JSON output of results which requires separate retrieval and script-based processing.

All integrated tools use a common XML TEI P5-based representation and interchange format, a packaged adaptation of the de facto Polish standard for linguistic stand-off description used in the National Corpus of Polish (Przepiórkowski et al., 2012). The format allows for representing layers of annotation,

various levels of ambiguity and disambiguation choices; below we present a fragment of morphosyntactic description of the word „dni" (En. *days*):

```
<seg xml:id="p4.s12.seg5" corresp="segmentation.xml#p4.seg221">
  <fs type="morph">
    <f name="orth">
      <string>dni</string>
    </f>
    <f name="disamb">
      <fs type="tool_report">
        <f fVal="#p4.s12.seg5.lex1.msd1" name="choice"></f>
        <f name="interpretation">
          <string>dzień:subst:pl:acc:m3</string>
        </f>
      </fs>
    </f>
  </fs>
</seg>
```

3 Motivation and initial experiments

Existing architectures such as the CLARIN Language Resource Switchboard (Zinn, 2016), WebLicht services (Hinrichs et al., 2010) or the Multiservice alone are not capable of carrying out complex processing combining linguistic analyses with text mining or advanced statistical calculations. Our solution to the problem would therefore be a combination of a general-purpose analytic tool and dedicated linguistic services. But before we begin it is worth veryfing whether RapidMiner out-of-the-box functionality would not be sufficient to complete this task. Since text analysis is one of the primary tasks for RapidMiner users, it may seem improbable that no existing configuration can be found to make it perform reliable linguistic analysis for Polish — definitely not an under-resourced language. Still, it occurs that Polish is much under-represented as far as integration of existing linguistic tools is concerned. The RapidMiner Studio offers various out-of-the-box linguistic analytic mechanisms but they are all incapable of processing Polish at a satisfactory level. Two type of tools are offered and were tested in this respect: integrated tools offered by the platform in the text processing extension and a proprietary text mining extension Rosette Text Toolkit.

The native processing provides standard filters for e.g. language-independent tokenization and stemming or stopword filtering for several languages available, but unfortunately not for Polish. For this reason it cannot be effectively used for any serious processing.

In turn, Rosette Text Toolkit is a multi-language solution offering processors for various linguistic tasks, including sentiment analysis, entity linking and many more. Even though Polish is one of the options in three basic components (Extract Sentences, Tokenization and Morphology), the toolkit seems to be unaware of specificities of Polish which makes it unacceptable in any application. To illustrate it in context of the task proposed later, Figure 2 presents results of the lemmatization and POS tagging of the first two lines of a linguistic poem „Słopiewnie" (Tuwim, 1971) famous for its neologisms. The results are highly unacceptable. Line 2 shows wrong lemma of a common noun present in contemporary dictionaries (should be lemmatized to *białodrzew*) while lines 4, 9 and 10 present various kind of problems for neologisms: the first and the last should be left as ignored words or analysed as verbs while the lemma assigned for the noun from line 9 is more than bizarre. In this case bad results are worse than none; it is not possible to distinguish new terms from wrong analyses of the dictionary terms neither turn off the lemma guessing mode.

These findings seem to prove that generic tools may not adapt to other languages easily since language specificities add to complexity of the task. At the same time linguistic analysis is successfully tackled with state-of-the-art linguistic tools available for individual languages — see Figure 3 for the same example analysed with Pantera tagger available both as offline application and as Web service.

Row No.	Token	Lemma	PartOfSpeech
1	W	w	ADP
2	białodrzewiu	białodrzewiu	NOUN
3	jaśnie	jasny	ADV
4	dźni	dźni	ADJ
5	słoneczko	słoneczko	NOUN
6	,	,	PUNCT
7	miodzie	miód	NOUN
8	złoci	złoć	NOUN
9	białopałem	białopałemo	NOUN
10	żyśnie	żyśna	NOUN

Figure 2: Results of POS tagging by Rosette Text Toolkit

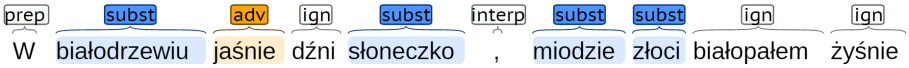

Figure 3: Results of POS tagging by the Multiservice

4 Interaction with the Multiservice

Interaction with Web services in RapidMiner can be achieved with ENRICH DATA BY WEBSERVICE operator[1] from the Web mining extension toolset. Request method (POST for Multiservice), request URL and the XML content of request body are set in processor parameters.

4.1 Web service request execution

The requests are being sent to `http://ws.multiservice.nlp.ipipan.waw.pl:80/WebService-1.0-SNAPSHOT/ClarinWS`. The listing below presents the structure of the initial `analyzeChain` request which commissions the work — in our case, processing the attribute *text* retrieved from an example with Pantera (lemmatizer and disambiguating tagger) and setting the output format of the result to be TEI P5:

```
<?xml version="1.0"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
            xmlns:n="http://ws.multiservice.zil.ipipan.waw.pl/">
  <s:Body>
    <n:analyzeChain>
      <text><%text%></text>
      <parts>
        <part>
          <serviceName>Pantera</serviceName>
        </part>
      </parts>
      <inputFormat>TEXT</inputFormat>
      <outputFormat>TEI</outputFormat>
    </n:analyzeChain>
  </s:Body>
</s:Envelope>
```

[1]Operators can be renamed to better illustrate their specific function but we use original names to facilitate re-creation of the process by the reader.

Due to asynchronous mode of execution of the Multiservice, the first interaction is intended only to begin the work and results in sending back the identifier of the task, which is a textual token. The TEI response must be retrieved separately when processing stops. Verification of processing status is possible with separate calls quoting the token:

```
<s:Body>
  <n:getStatus>
    <token><%token%></token>
  </n:getStatus>
</s:Body>
```

Querying for status returns `IN_PROGRESS` when processing is still running and `DONE` when it ended successfully[2]. Results can be retrieved in a similar manner:

```
<s:Body>
 <n:getResult>
   <token><%token%></token>
 </n:getResult>
</s:Body>
```

4.2 Modelling the status checking loop

LOOP UNTIL operator is used to periodically query for appropriate status of the linguistic processing. Figure 4 shows the details of the process: since processing components have access to examples and attributes only, the best method of interacting with the process is by passing values in the result set as input and output to subsequent operators. Only one attribute with a given name can exist, so the process removes *status* from the set (which contains a single row of data), adds the value of the current status as a new *status* attribute and filters examples leaving only those with `IN_PROGRESS` status which results in an empty dataset when processing has finished vs. a single row when it is still in progress. Then PERFORMANCE operator is used to count the rows in the result set and pass the value to the parent processor to stop execution when the result set contains any rows. A delay of 100 ms is introduced to limit the number of requests (although it could be removed to increase performance).

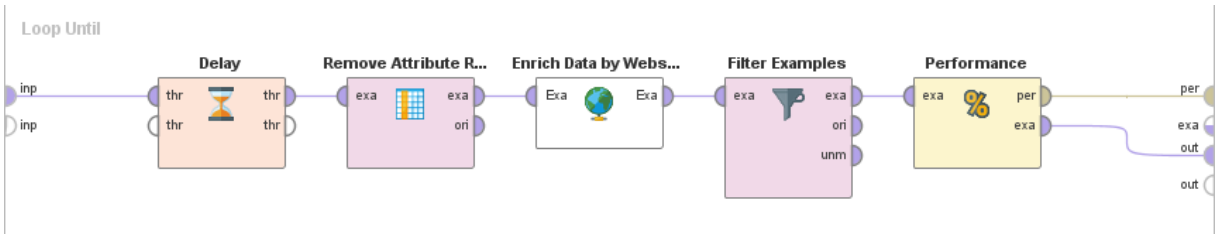

Figure 4: Using the LOOP UNTIL operator to wait for completion of the linguistic analysis

5 Sample analysis: frequency of unknown words in a text set

The process can be easily modelled within a LOOP FILES operator which, for each file retrieved from a folder, reads the document, converts it to internal representation of text and creates example set from text; then retrieves the results of the Web service execution and extracts parts of data relevant for further processing. After execution of the process the result sets from all partial results can be aggregated, selected attributes grouped and their frequencies counted.

The demonstration of such setting was used to prepare a list of unknown words in a set of texts corresponding to chapters from Solaris (Lem, 1970), a novel by Stanisław Lem, Polish science-fiction writer famous for his lexical creativity. This mini-corpus was composed of UTF-8-encoded plain text files with a total of 57K words.

The configuration of operators (with LOOP UNTIL detailed in Figure 4 and embedded process LOOP FILES integrated as sub-box to maintain single-picture view) is presented in Figure 5. Files in a folder

[2]There are also other statuses for different situations, please see the Multiservice description at `http://zil.ipipan.waw.pl/Multiservice`.

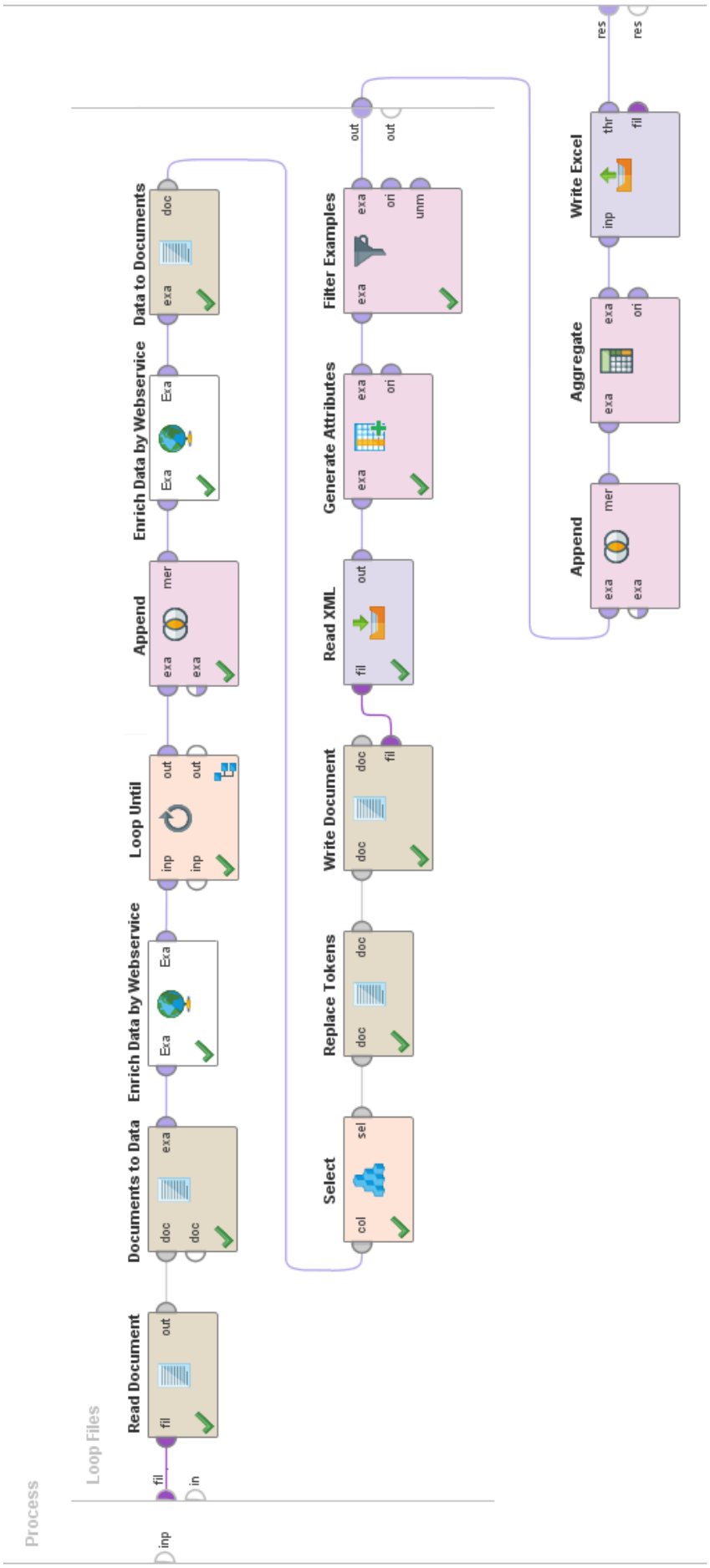

Figure 5: The multi-level setting of operators for calculating frequency of unknown words in a text set

are browsed and the Web service is called for the text content of each file. Pantera tagger processes the input and generates three layers of annotation: sentence segmentation, tokenization and disambiguated morphosyntax. Result sets are created from the XML TEI content with READ XML operator which extracts morphosyntactic interpretations selected by the tagger by evaluating an XPath expression `//f[@name='interpretation']/string` (cf. TEI excerpt in Section 2.2). The result string is then used as a source for two regular expressions which generate new attributes corresponding to lemma and POS tag of each token. Finally for each file, the example set is filtered to keep only rows with `tag` value equal to `ign` which corresponds to an unknown word[3]. Outside the loop the results are appended to form a single example set and then aggregated by counting frequencies of distinct lemmata. At the end, the result is exported to an Excel file.

The setting can look complex at first glance yet it can be easily reused: for researchers familiar with the NKJP format it would be sufficient to update path expressions to extract different parts of the linguistic analysis provided by the Web service. More advanced users can experiment with the platform, investigate capabilities of other available operators and create different data flows.

The result of processing of Solaris showed 278 unrecognized words (Figure 6 presenting the top of the list) which could be easily further categorized by the researcher.

Row No.	lemma	count(lemma) ↓
20	Harey	11
14	Gibarian	7
30	Sartorius	5
32	Snaut	4
15	Gibariana	3
18	Gravinsky	2
33	Snauta	2
54	mimoid	2
55	mimoidu	2
67	solariańskiej	2

Figure 6: Unrecognized orthographic forms with frequencies greater than one in Lem's Solaris — the result of the experimental setting

6 Conclusions

Availability of data mining tools and growing supply of linguistic Web services offered non-expert users new methods of combining resources and tools to perform their analytical tasks. In this respect our approach seems to go in line with requirements of the humanities, rarely interested in complex installation or configuration of software.

The experimental setting could be improved in many ways, e.g. by including process branches depending on Web service execution status (which can fail), adaptation of the setting to maximum request size allowed by the server or application of sophisticated analytical mechanisms offered by RapidMiner.

Since the motivation of the attempt was to illustrate how linguistic processing for Polish can be integrated into a larger environment rather than optimize it for performance, there are obviously more efficient methods for carrying out the same simple task, with components running locally and extraction

[3]Early filtering helps maintain the 10,000-row limit for the community version of Rapidminer.

scripts implemented in expressive programming languages. Still, the setting offers a valuable playground for non-technical researchers and might be used to raise their interest in further exploration of more advanced analytical tools.

Acknowledgements

The work reported here was financed as part of the investment in the CLARIN-PL research infrastructure funded by the Polish Ministry of Science and Higher Education.

References

Szymon Acedański. 2010. A Morphosyntactic Brill Tagger for Inflectional Languages. In Hrafn Loftsson, Eiríkur Rögnvaldsson, and Sigrún Helgadóttir, editors, *Advances in Natural Language Processing*, volume 6233 of *Lecture Notes in Computer Science*, pages 3–14. Springer.

Aleksander Buczyński and Aleksander Wawer. 2008. Shallow parsing in sentiment analysis of product reviews. In Sandra Kübler, Jakub Piskorski, and Adam Przepiórkowski, editors, *Proceedings of the LREC 2008 Workshop on Partial Parsing: Between Chunking and Deep Parsing*, pages 14–18, Marrakech. ELRA.

Adam Dudczak. 2007. Zastosowanie wybranych metod eksploracji danych do tworzenia streszczeń tekstów prasowych dla jezyka polskiego (En. Application of selected data exploration methods to summarization of Polish newspaper articles). MSc thesis.

Erhard W. Hinrichs, Marie Hinrichs, and Thomas Zastrow. 2010. Weblicht: Web-based LRT services for german. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, System Demonstrations*, pages 25–29. The Association for Computer Linguistics.

Lisa Kart, Gareth Herschel, Alexander Linden, and Jim Hare. 2016. Magic Quadrant for Advanced Analytics Platforms. Technical report, Gartner.

Łukasz Kobyliński. 2014. PoliTa: A multitagger for Polish. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*, pages 2949–2954, Reykjavík, Iceland. ELRA.

Mateusz Kopeć. 2016. Nicolas Summarizer. [on-line] `http://zil.ipipan.waw.pl/Nicolas`.

Stanisław Lem. 1970. *Solaris*. A Harvest book. Harcourt.

Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. 2006. YALE: rapid prototyping for complex data mining tasks. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*, pages 935–940. ACM.

Maciej Ogrodniczuk and Mateusz Kopeć. 2011. Rule-based coreference resolution module for Polish. In *Proceedings of the 8th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2011)*, pages 191–200, Faro, Portugal.

Maciej Ogrodniczuk and Michał Lenart. 2012. Web Service integration platform for Polish linguistic resources. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012*, pages 1164–1168, Istanbul, Turkey. ELRA.

Maciej Ogrodniczuk, Katarzyna Głowińska, Mateusz Kopeć, Agata Savary, and Magdalena Zawisławska. 2015. *Coreference in Polish: Annotation, Resolution and Evaluation*. Walter De Gruyter.

Adam Przepiórkowski and Aleksander Buczyński. 2007. Spejd: Shallow Parsing and Disambiguation Engine. In Zygmunt Vetulani, editor, *Proceedings of the 3rd Language & Technology Conference*, pages 340–344, Poznań, Poland.

Adam Przepiórkowski, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk, editors. 2012. *Narodowy Korpus Jezyka Polskiego (En. National Corpus of Polish; in Polish)*. Wydawnictwo Naukowe PWN, Warsaw.

Adam Radziszewski and Tomasz Śniatowski. 2011. A memory-based tagger for Polish. In Zygmunt Vetulani, editor, *Proceedings of the 5th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 29–36, Poznań, Poland.

Adam Radziszewski. 2013. A tiered CRF tagger for Polish. In R. Bembenik, Ł. Skonieczny, H. Rybiński, M. Kryszkiewicz, and M. Niezgódka, editors, *Intelligent Tools for Building a Scientific Information Platform: Advanced Architectures and Solutions*. Springer Verlag.

Nadav Rotem. 2003. The Open Text Summarizer. [on-line] `http://libots.sourceforge.net/`.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. BRAT: a web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL'12, pages 102–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Joanna Świetlicka. 2010. Metody maszynowego uczenia w automatycznym streszczaniu tekstów (En. Machine learning methods in automatic text summarization; in Polish). Master's thesis, Warsaw University, Poland.

Julian Tuwim. 1971. *Wiersze zebrane*. Wiersze zebrane. Czytelnik.

Jakub Waszczuk, Katarzyna Głowińska, Agata Savary, Adam Przepiórkowski, and Michał Lenart. 2013. Annotation tools for syntax and named entities in the National Corpus of Polish. *International Journal of Data Mining, Modelling and Management*, 5(2):103–122.

Jakub Waszczuk. 2012. Harnessing the CRF complexity with domain-specific constraints. The case of morphosyntactic tagging of a highly inflected language. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 2789–2804, Mumbai, India.

Alina Wróblewska. 2014. *Polish Dependency Parser Trained on an Automatically Induced Dependency Bank*. Phd thesis, Institute of Computer Science, Polish Academy of Sciences, Warsaw.

Claus Zinn. 2016. The CLARIN Language Resource Switchboard. [on-line] `https://www.clarin.eu/sites/default/files/zinn-CLARIN2016_paper_26.pdf`.