WCCL Relation — a Toolset for Rule-based Recognition of Semantic Relations Between Named Entities

Michał Marcińczuk
Wrocław University of Technology
michal.marcinczuk@pwr.wroc.pl

Abstract

In this paper we cover the problem of recognition of semantic relations between proper names (PNs) in running text. We focus on the manual rule creation approach and discuss to what extent the existing tools can be used for this task. As a result of our initial research we developed a rule-based toolset for recognition of relations between PNs called WCCL Relation. The toolset is built on the top of WCCL Match — a language for text annotation, which is a part of a WCCL framework (an open source, released under the GNU LGPL 3.0). The WCCL Relation toolset is language independent and can be used for almost any natural language and language tagset. We present several use cases and sample rules for recognition of semantic relations in Polish texts.

1 Introduction

Recognition of semantic relations between named entities is one of the information extraction major tasks. Its goal is to identify pairs of named entities (text fragments) connected other by a semantic relation on the basis of their context. In the majority of approaches the named entities are recognised beforehand and the task is limited to discovering and categorising connections between those entities. The list of possible relation categories is unbounded and it depends on the desired application, the scope of the named entities and the available resources. For example Marcińczuk and Ptak (2012) defined 8 coarse-grained categories of semantic relations (location, origin, nationality, affiliation, neighbourhood, creator, composition and alias). In turn Linguistic Data Consortium (2008) defined a set of 8 general relations (i.e., physical, part-whole, personal-social, organization-affiliation, agent-artifact and general-affiliation) with several subcategories. In the bioinformatic domain there are two common categories of relations between genes, proteins and associated entities — *protein-component* and *subunit-complex* (Pyysalo et al., 2011).

There are two main approaches to relation recognition — construction of human-readable rules and construction of statistical models (machine learning). According to Jiang (2012) the most common approach is the one based on the statistical models. There are also several rule-based approaches, like manual rule creation (Marciniak and Mykowiecka, 2007; Santos et al., 2010; Abacha and Zweigenbaum, 2011) and rule induction (Feldman et al., 2006; Brun and Hagège, 2009). The low interest in developing rule-based systems might be caused by a lack of robust and accessible tools for rule construction and execution. For example, the well-known general framework GATE (Cunningham et al., 2011) does not support relation recognition within its rule formalism JAPE (Cunningham et al., 2000).

Despite the manual rule creation is less popular than the statistical approaches in the task of relation recognition, the rule-based approaches have several advantages over statistical-based approaches. The first one is the traceability and full control on decisions made by the system. The other one is the ease in manual tuning for new types of text. The last but not least it does not require annotated data.

In this paper we investigate the problem of rule creation for recognition of semantic relations between proper names. We present a language independent formalism for rule creation and execution called *WCCL Relation*. The language is built on top of an open-source framework WCCL (Radziszewski et al., 2011). We present several use cases of the language applications in the context of recognition of semantic relations between proper names for Polish.

Proceedings of Recent Advances in Natural Language Processing, pages 436–442,
Hissar, Bulgaria, 7-13 September 2013.

2 Related works

Before we decided to create the WCCL Relation toolset we had considered several existing approaches described in the literature. Abacha and Zweigenbaum (2011) used a custom rule notion and a software to develop a set of rules for their task. However, the main emphasis was put on the task definition and discussion of its difficulties. Less effort was made to create a general solution that would result in an universal system or formalism for rule creation and execution.

There is another group of works utilizing the Xerox Incremental Parser (XIP). According to Aït-Mokhtar et al. (2002), XIP is a formalism which allows to recognise n-ary linguistic relations between words or constituents on the basis of global or local structural, topological and/or lexical conditions. Brun and Hagège (2009) used the formalism in semi-supervised rule creation (the rules were used to recognise Olympic games events). Santos et al. (2010) used XIP to create rules for recognition of family relations between people. Despite the formalism looks very promising the distribution and licensing is not clear and the XIP implementation is not freely available.

There is also another system called TEG (Feldman et al., 2006) which offers a stochastic context-free grammar (SCFG) to write rules for recognition of relations between named entities. The system offers a semi-supervised method for rule creation. Unfortunately, according to our best knowledge the system is not publicly accessible.

An open-source Python platform for text processing called NLTK[1] (Bird et al., 2009) provides a simple tool to relation recognition based on regex patterns. The patterns are tested against a plain text enriched with part of speech tags. This approach can be suitable for many simple uses cases but it is troublesome to use for languages with rich morphology (each word is described by a set of morphological attributes, not only by the part of speech tag). It also does not support multi-layered semantic annotations.

Taking into consideration the above solutions we decided to construct a customized toolset for the rule-based relation recognition utilizing an existing open-source framework for text matching and annotation. The following section presents the current version of WCCL Relation toolset.

[1] `http://nltk.org`

3 WCCL Relation

WCCL Relation is a toolset designed for a rule-based recognition of relations between pairs of annotations within a sentence in a morphologically tagged and semantically annotated texts. Its grammar is based on the WCCL Match[2] (Marcińczuk and Radziszewski, 2013) and extends it by a new operator for relation creation. A WCCL Relation rule consists of three sections. The first section (`match`) contains a set of operators used to match a sequence of tokens and annotations (named entities, chunks, etc.). The second section (`cond`) is optional and contains a set of additional conditions which must be satisfied by the matched elements. The last section (`actions`) contains a set of operators to be performed on the matched elements. Comparing to the original WCCL Match grammar, the WCCL Relation grammar contains an additional operator called `link` which allows to create a connection of given category between two matched elements. Below is a sample rule which matches a sequence "PERSON born in CITY" and creates a connection between the PERSON and the CITY names of type *origin*.

```
apply(
  match(
    // match annotation of type person_nam
    is("person_nam"),              // group 1
    // match word with base form 'born'
    equal( base[0], "urodzić"),  // group 2
    equal( base[0], "się"),      // group 3
    // match word with base form 'in'
    equal( base[0], "w"),        // group 4
    // match annotation of type city_nam
    is("city_nam"),              // group 5
  ),
  actions(
    link(1, "person_nam", 5, "city_nam", "origin")
))
```

WCCL Match offers a set of operators for matching a sequence of elements. Below is a list of operators used in the examples presented in the article[3]:

- `is(type)` — matches an annotation of given type,
- `equal(base[0], value)` — matches a token with a base form equal to *value*,
- `inter(base[0], values)` — matches a token with a base form present in the array of values,

[2] Part of WCCL framework (Radziszewski et al., 2011)

[3] The complete list of the WCCL Match operators can be found in Marcińczuk and Radziszewski (2013).

- `repeat(op)` — matches a sequence of elements matching the *op* operator,
- `not(op)` — matches a token not matching the *op* operator,
- `isannpart(0, type)` — matches a token that is a part of an annotation of given type,
- `and(op1,op2,..,opn)` — matches a token if all operators are valid,
- `oneof(variant1,...,variant2)` — matches a sequence of elements for the first valid variant,
- `annsub(token, type)` — test if given token is part of an annotation of given type,
- `agrpp(word1, word2)` — test agreement of two given words,
- `outside(index)` — test if given token index is inside sentence boundary, can be used to test if given token is the first or the last token in the sentence,

The execution of a WCCL Relation rule consists of three steps (all of them are transparent to user). In the first step, the WCCL Relation rule is transformed into a WCCL Match rule. In this step all the additional operators are transformed to operators valid for WCCL Match. In the second step, the WCCL Match rule is run on a given text. In the last step, the result of matching (set of annotations) is interpreted and transformed into a set of relations.

Below is the result of transformation the WCCL Relation rule to the WCCL Match rule. Here, the `link` operator was replaced with a set of three `match` operators.

```
apply(
  match(
    // match annotation of type person_nam
    is("person_nam"),
    // match word with base form 'born'
    equal( base[0], "urodzić"),
    equal( base[0], "się"),
    // match word with base form 'in'
    equal( base[0], "w"),
    // match annotation of type city_nam
    is("city_nam"),
  ),
  actions(
    mark(:1, :5, "relation.origin.r1"),
    mark(:1, "relation.origin.r1.person_nam"),
    mark(:5, "relation.target.r1.city_nam")
))
```

4 Case studies

In this section we present several use cases already covered by the WCCL Relation toolset. We assumed that the proper names were recognised beforehand using an external tool. For Polish we used a tool called Liner2[4] (Marcińczuk et al., 2013) with a model for 56 categories of proper names.

4.1 Auxiliary annotations

The standard WCCL Match operator `mark` can be used to introduce the auxiliary annotations which can be referenced by other rules. This simplifies the final rules recognising the relations. For example, a common action is to ignore phrases in parentheses which can separate two named entities. This can be done using the following rule. The rule matches a text that is delimited by a pair of elements: "(" and ")" or "[" and "]".

```
apply(
  match(
    oneof(
      variant(
        in("(", base[0]),
        repeat(not(inter(base[0], [")", "("]))),
        in(")", base[0])
      ),
      variant(
        in("[", base[0]),
        repeat(not(inter(base[0], ["]", "["]))),
        in("]", base[0])
      )
    )
  ),
  actions(
    mark(M, "parentheses")
))
```

The following rule extends the previous rule recognising the *origin* relation between a person name and a city name by including an optional phrase in parentheses after the person name.

```
apply(
  match(
    // match annotation of type person_nam
    is("person_nam"),              // group 1
    // match optional phrase in parentheses
    optional(is("parentheses"))    // group 2
    // match word with base form 'born'
    equal( base[0], "urodzić"),    // group 3
    equal( base[0], "się"),        // group 4
    // match word with base form 'in'
    equal( base[0], "w"),          // group 5
    // match annotation of type city_nam
    is("city_nam"),                // group 6
  ),
  actions(
    link(1, "person_nam", 6, "city_nam", "origin")
))
```

4.2 Possessive named entities

The following rule is a naïve rule for recognition of a *location* relation between a person name and

[4] http://nlp.pwr.wroc.pl/liner2

a city name (i.e. a person is in a city).

```
apply(
  match(
    is("person_nam"),              // group 1
    // match word with base form 'in'
    in("w", base[0]),              // group 2
    is("city_nam")                 // group 3
  ),
  actions(
    link(1, "person_nam", 3, "city_nam", "location")
))
```

However this rule is not always true. For example when the person name is an possessive argument of an other subject then the relation does not occur between the person name and the city name but between the possessive phrase and the city name. Consider the following sentence: *Pomnik Wojtyły w Krakowie* (eng. *Wojtyła monument in Kraków*). In the sentence it is stated that the *monument* is located in Kraków and it does not mean that *Wojtyła* is also in Kraków. In order to handle properly such situations we must recognise the possessive nouns. This can be done with the following rule. This rule test a person name preceded by a noun. If the person name and the noun do not agree in case then the person name is being recognised as possessive phrase.

```
apply(
  match(
    in(subst, class[0]),
    is("person_nam")
  ),
  cond(
    in(subst, class[first(:2)]),
    not(agrpp(first(:1), first(:2), {cas}))
  ),
  actions(
    mark(M, "possessive")
))
```

Now we can add a condition in the `cond` section to ignore the person names which are part of a possessive phrase. Below is the original rule with the mentioned condition.

```
apply(
  match(
    is("person_nam"),          // group 1
    in("w", base[0]),          // group 2, eng. "in"
    is("city_nam")             // group 3
  ),
  cond(
    not(annsub(:1, "possessive"))
  ),
  actions(
    link(1, "person_nam", 3, "city_nam", "location")
))
```

4.3 Multiple relations

The other common situation is recognition of multiple relations within a single matched sequence. Below is a sample rule which matches the sequence "COUNTRY (CITY and CITY)" and creates two links: both city names are connected with the country name as separate relations.

```
apply(
  match(
    is("country_nam"),         // group 1
    inter(base[0], "("),       // group 2
    is("city_nam"),            // group 3
    inter(base[0], "i"),       // group 4
    is("city_nam"),            // group 5
    inter(base[0], ")"),       // group 6
  ),
  actions(
    link(1,"country_nam",3,"city_nam","location"),
    link(1,"country_nam",5,"city_nam","location")
))
```

4.4 Detecting sentences containing only two annotations

In some cases when there are only two proper names of given categories in a sentence, the proper names can be connected with a certain relation category no matter of their context. For example, in most case a road name and a city name preceded by a preposition *in* are connected with a *location* relation. Below is an auxiliary rule that matches the text fragments not annotated with a road name nor a city name.

```
apply(
  match(
    repeat(
      and(
        not(isannpart(0,"road_nam")),
        not(isannpart(0,"city_nam"))
      )
    )
  ),
  actions(
    mark(M, "not_road_city")
))
```

Using the above auxiliary annotation `not_road_city` we can construct the following rule (the `cond` is used to check if the matched sequence spans over a whole sentence).

```
apply(
  match(
    is("not_road_city"),       // group 1
    is("road_nam"),            // group 2
    is("not_road_city"),       // group 3
    in("w", base[0]),          // group 4, eng. ``in''
    is("city_nam"),            // group 5
    is("not_road_city")        // group 6
  ),
  cond(
    outside(first(M) - 1), outside(last(M) + 1)
  ),
  actions(
    link(2, "road_nam", 4, "city_nam", "location")
))
```

5 Evaluation

In the evaluation we used the KPWr corpus (Broda et al., 2012)[5], which is the only available corpus annotated with semantic relations between proper names for Polish. We followed the evaluation procedure presented by Marcińczuk and Ptak (2012),

[5] http://nlp.pwr.wroc.pl/kpwr.

where the corpus was divided into three parts: train, tune and test. The train and tune parts were used for the rule development and the test part for the final performance comparison.

As the work is still in progress we started from the most numerous relation category in KPWr that is *location* (about 800 relations). The current set contains 34 rules (6 of them are auxiliary rules). It took about 6 hours to develop the rules. The set covers almost 40% of *location* relations in the train part, 30% in the tune part and 22% in the test part with the precision between 87–90%. The recall is low but in terms of F-measure the results are comparable with the results obtained for the statistical methods presented by Marcińczuk and Ptak (2012). On the test part the statistical model obtained 36.09% F-measure with 31.20% precision, while the manually crafted rules obtained already 34.97% F-measure with 87.18% precision. Higher precision is more useful for processing large volumes of texts where recall is not an issue. Our final goal is to construct a set of rules covering all categories of semantic relations present in the KPWr corpus.

6 WCCL Relation is language independent

Since the WCCL framework is language independent, also WCCL Relation is language independent. Note, that the rules written for one language are not directly usable for other languages. They can be adopted to another language or tagset but they have to be anywise translated.

WCCL Relation can be used to process any language which tagset conforms the following requirements:

- the tagset defines a non-empty set of grammatical classes and possibly empty set of attributes;
- each grammatical class is assigned a set of attributes that are required for the class and a set of optional attributes;
- each attribute is assigned a set of its possible values;
- mnemonics used for grammatical classes and attribute values are unique;
- and the tags are represented as a string of comma-separated mnemonics.

7 Input/output format

The WCCL Relation rules can be executed in two ways: in a console to process an XML file in the CCL format or in a code using the API.

7.1 Processing CCL files

Below is a sample XML in the CCL format for a sentence “Eiffel Tower is located in Paris”. The file contains morphological tags for each word and semantic annotations (`facility_nam` for *Eiffel Tower* and `city_nam` for *Paris*).

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE chunkList SYSTEM "ccl.dtd">
<chunkList>
 <chunk type="p" id="ch1">
  <sentence id="s1">
   <tok>
    <orth>Wieża</orth> <!-- Tower : facility_nam -->
    <lex disamb="1"><base>wieża</base>
      <ctag>subst:sg:nom:f</ctag></lex>
    <ann chan="city_nam">0</ann>
    <ann chan="facility_nam">1</ann>
   </tok>
   <tok>
    <orth>Eiffla</orth> <!-- Eiffel : facility_nam-->
    <lex disamb="1"><base>Eiffel</base>
      <ctag>subst:sg:gen:m1</ctag></lex>
    <ann chan="city_nam">0</ann>
    <ann chan="facility_nam">1</ann>
   </tok>
   <tok>
    <orth>znajduje</orth> <!-- is located -->
    <lex disamb="1"><base>znajdować</base>
      <ctag>fin:sg:ter:imperf</ctag></lex>
    <lex disamb="1"><base>znajdywać</base>
      <ctag>fin:sg:ter:imperf</ctag></lex>
    <ann chan="city_nam">0</ann>
    <ann chan="facility_nam">0</ann>
   </tok>
   <tok>
    <orth>się</orth>
    <lex disamb="1"><base>się</base>
      <ctag>qub</ctag></lex>
    <ann chan="city_nam">0</ann>
    <ann chan="facility_nam">0</ann>
   </tok>
   <tok>
    <orth>w</orth> <!-- in -->
    <lex disamb="1"><base>w</base>
      <ctag>prep:loc:nwok</ctag></lex>
    <ann chan="city_nam">0</ann>
    <ann chan="facility_nam">0</ann>
   </tok>
   <tok>
    <orth>Paryżu</orth> <!-- Paris : city_nam -->
    <lex disamb="1"><base>Paryż</base>
      <ctag>subst:sg:loc:m3</ctag></lex>
    <ann chan="city_nam">1</ann>
    <ann chan="facility_nam">0</ann>
   </tok>
  </sentence>
 </chunk>
</chunkList>
```

Below is an XML output generated by the tool containing a single semantic relation of type *location* between *Eiffel Tower* and *Paris*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE chunkList SYSTEM "ccl.dtd">
<relations>
 <rel name="location" set="Syntactic relations">
   <from sent="s1" chan="facility_nam">1</to>
   <to sent="s1" chan="city_nam">1</from>
  </rel>
</relations>
```

7.2 Using API

The WCCL Relation tool provides set of API functions in Python to execute the rules directly in the code. Below we present a very brief description of the API. More information and examples can be found on the following page: `http://nlp.pwr.wroc.pl/wccl-relation`. The API provides the following functions:

- `process_file(filepath)` — process a single CCL file,
- `process_files(filepaths)` — process a set of CCL files,
- `process_sentence(sentence)` — process a single sentence represented as an object of class `corpus2.AnnotatedSentence` [6],
- `process_document(document)` — process a single document represented as on object of class `corpus2.DocumentPtr`[6].

All the presented functions return a set of objects of class `corpus2.Relation`[6] representing the recognised relations.

8 Conclusion and future work

In the paper we presented a result of ongoing work on creation a language independent rule-based toolset for recognition of relations between named entities, called WCCL Relation. The toolset is build on the top of an open-source framework called WCCL. A set of use cases for recognition of semantic relations between proper names for Polish was presented.

WCCL Relation is build on the top of an open-source framework called WCCL which is implemented in C++ and its source code is released under GNU LGPL 3.0[7]. WCCL Relation has a form of a Python script that is also released under the same license[8].

The described work is still in progress. On one hand we are still working on a set of rules for recognition of 8 categories of semantic relations between PNs for Polish. On the other hand we are still extending the WCCL Relation toolset with new features. One of the planned features is a support for names enumerations. The other are access to word dependency features, tests on distance between matched elements and support for relations between nested annotations.

[6] `http://nlp.pwr.wroc.pl/redmine/projects/corpus2/wiki`

[7] `http://www.nlp.pwr.wroc.pl/wccl`.

[8] `http://nlp.pwr.wroc.pl/wccl-relation`.

Acknowledgments

The work was funded by the NCBiR NrU.: SP/I/1/77065/10.

References

Asma Ben Abacha and Pierre Zweigenbaum. 2011. Medical entity recognition: a comparison of semantic and statistical methods. In *Proceedings of BioNLP 2011 Workshop*, BioNLP '11, pages 56–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

S. Aït-Mokhtar, J.-P. Chanod, and C. Roux. 2002. Robustness beyond shallowness: incremental deep parsing. *Natural Language Engineering*, 8(3):121–144, June.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition.

Bartosz Broda, Michał Marcińczuk, Marek Maziarz, Adam Radziszewski, and Adam Wardyński. 2012. KPWr: Towards a Free Corpus of Polish. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Caroline Brun and Caroline Hagège. 2009. Semantically-Driven Extraction of Relations between Named Entities. *Research in Computing Science*, 41:35–46.

Hamish Cunningham, Diana Maynard, and Valentin Tablan. 2000. JAPE: a Java Annotation Patterns Engine. Technical Report CS—00—10, University of Sheffield, Department of Computer Science.

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, Valentin Tablan, Niraj Aswani, Ian Roberts, Genevieve Gorrell, Adam Funk, Angus Roberts, Danica Damljanovic, Thomas Heitz, Mark A. Greenwood, Horacio Saggion, Johann Petrak, Yaoyong Li, and Wim Peters. 2011. *Text Processing with GATE (Version 6)*. University of Sheffield Department of Computer Science.

Ronen Feldman, Benjamin Rosenfeld, and Moshe Fresko. 2006. TEG — a hybrid approach to information extraction. *Knowledge and Information Systems*, 9(1):1–18, January.

Jing Jiang. 2012. Information Extraction from Text. In Charu C. Aggarwal and Cheng Xiang Zhai, editors, *Mining Text Data*, pages 11–41. Springer.

Linguistic Data Consortium. 2008. ACE (Automatic Content Extraction) English Annotation Guidelines for Relations (Version 6.2).

Michał Marcińczuk and Marcin Ptak. 2012. Preliminary Study on Automatic Induction of Rules for Recognition of Semantic Relations between Proper Names in Polish Texts. In Petr Sojka, Aleš Horák, Ivan Kopecek, and Karel Pala, editors, *Text, Speech and Dialogue, 15th International Conference, TSD 2012, Brno, Czech Republic, September 3-7, 2012. Proceedings*, volume 7499 of *Lecture Notes in Computer Science*, pages 264–271. Springer Berlin Heidelberg.

Małgorzata Marciniak and Agnieszka Mykowiecka. 2007. Automatic processing of diabetic patients' hospital documentation. *Annual Meeting of the ACL*, pages 35–42.

Michał Marcińczuk and Adam Radziszewski. 2013. Wccl match – a language for text annotation. In MieczysławA. Kłopotek, Jacek Koronacki, Małgorzata Marciniak, Agnieszka Mykowiecka, and SławomirT. Wierzchoń, editors, *Language Processing and Intelligent Information Systems*, volume 7912 of *Lecture Notes in Computer Science*, pages 131–144. Springer Berlin Heidelberg.

Michał Marcińczuk, Jan Kocoń, and Maciej Janicki. 2013. Liner2 – a customizable framework for proper names recognition for polish. In Robert Bembenik, Lukasz Skonieczny, Henryk Rybinski, Marzena Kryszkiewicz, and Marek Niezgodka, editors, *Intelligent Tools for Building a Scientific Information Platform*, volume 467 of *Studies in Computational Intelligence*, pages 231–253. Springer Berlin Heidelberg.

Sampo Pyysalo, Tomoko Ohta, and Jun'ichi Tsujii. 2011. Overview of the Entity Relations (REL) supporting task of BioNLP Shared Task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 83–88, Portland, Oregon, USA. Association for Computational Linguistics.

Adam Radziszewski, Adam Wardyński, and Tomasz Śniatowski. 2011. WCCL: A Morpho-syntactic Feature Toolkit. In Ivan Habernal and Václav Matoušek, editors, *Text, Speech and Dialogue, 14th International Conference, TSD 2011, Pilsen, Czech Republic, September 1-5, 2011. Proceedings*, volume 6836 of *Lecture Notes in Computer Science*, pages 434–441. Springer Berlin Heidelberg.

Daniel Santos, Nuno Mamede, and Jorge Baptista. 2010. Extraction of Family Relations between Entities. In Luıs S. Barbosa and Miguel P. Correia, editors, *Proceedings of INForum 2010 - II Simposio de Informatica*, pages 549–560.