Detecting Duplicates with Shallow and Parser-based Methods

Sven HARTRUMPF
IICS, FernUniversität in Hagen, Hagen, Germany
sven.hartrumpf@fernuni-hagen.de

Tim VOR DER BRÜCK
IICS, FernUniversität in Hagen, Hagen, Germany
tim.vorderbrueck@fernuni-hagen.de

Christian EICHHORN
Informatik 1, TU Dortmund, Dortmund, Germany
christian.eichhorn@tu-dortmund.de

[1]

Abstract:

Identifying duplicate texts is important in many areas like plagiarism detection, information retrieval, text summarization, and question answering. Current approaches are mostly surface-oriented (or use only shallow syntactic representations) and see each text only as a token list. In this work however, we describe a deep, semantically oriented method based on semantic networks which are derived by a syntactico-semantic parser. Semantically identical or similar semantic networks for each sentence of a given base text are efficiently retrieved by using a specialized semantic network index. In order to detect many kinds of paraphrases the current base semantic network is varied by applying inferences: lexico-semantic relations, relation axioms, and meaning postulates. Some important phenomena occurring in difficult-to-detect duplicates are discussed. The deep approach profits from background knowledge, whose acquisition from corpora like Wikipedia is explained briefly. This deep duplicate recognizer is combined with two shallow duplicate recognizers in order to guarantee high recall for texts which are not fully parsable. The evaluation shows that the combined approach preserves recall and increases precision considerably, in comparison to traditional shallow methods. For the evaluation, a standard corpus of German plagiarisms was extended by four diverse components with an emphasis on duplicates (and not just plagiarisms), e.g., news feed articles from different web sources and two translations of the same short story.

Keywords:

Duplicate detection, Plagiarism, Semantic networks, Support vector machine, Paraphrases, Entailments

[1] **978-1-4244-6899-7/10/$26.00 ©2010 IEEE**

1. Introduction

With the growth of the web the number of available texts has increased rapidly. The number of duplicates increased with similar speed, deliberately by generating plagiarisms or unwittingly by presenting information already given by other users or services (e.g., in Weblogs, Microblogs and News, cf. [1, p. 82]).

To detect duplicates is a relevant task for many different areas: applications regarding information access like search engines or question answering systems try not to response with duplicate information to user requests. Copyright owners and tutors want to find cases of copyright violations and plagiarism even if the violator used techniques to obfuscate the source. Other uses could include backup tools trying to eschew redundant files or computer administrators searching for redundant files which can be deleted in order to save disk space.

In prior work, duplicate detection employs shallow methods, working on surface-oriented factors or features only, which are mainly derived from n-grams, rare words and spelling errors, with n-grams being used most frequently [2, Section 3.2]. Even if the capability of these approaches has increased, they are still capable of detecting only three quarters of the tested plagiarisms [3].

Using the semantics of words, sentences, paragraphs, or even whole texts, two texts which are semantic duplicates, i.e., expressing the same content without sharing many words or word sequences and hence without having similar values of shallow features, can be tackled. Since shallow checkers can easily be tricked by experienced users which employ advanced paraphrase techniques, a deep approach that compares full semantic representations of two given texts is designed, implemented, and evaluated in the SemDupl (Semantic

Duplicate) project in order to detect even obfuscated plagiarisms and semantic duplicates.

Our deep duplicate checker operates on semantic networks (SNs) following the MultiNet formalism [4]. These networks are derived from text by a deep syntactico-semantic parsing based on a word-class functional analysis. Such an SN consists of nodes representing concepts (disambiguated word readings) and arcs denoting relations (or functions) between concepts. An example SN is shown in Figure 1. Important MultiNet relations/functions are:

- *DIFF: Function specifying the difference of sets
- *ITMS: Function enumerating a set
- PRED: Predicative concept characterizing a plurality

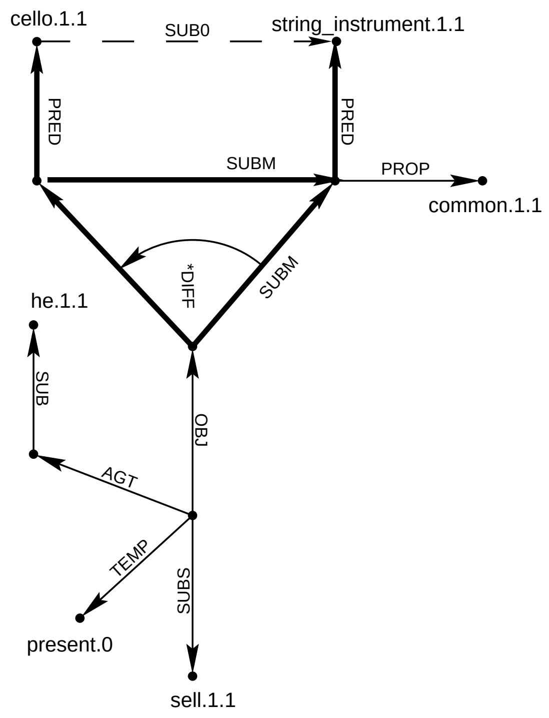

Figure 1. Matching a deep pattern to an SN representing the sentence: *"He sells all common string instruments except celli."* The matching edges are printed in bold, the dashed line is the inferred edge.

- PROP: Relation between object and property
- SUB/SUB0: Relation of subordination for conceptual objects (hyponym/ instance of)
- SUBM: Set inclusion

2. State of the Art

As stated above, detecting duplicates is of high interest for holders of rights and tutors. Therefore, many tools exist to detect plagiarisms in given corpora or the web. Below are some of the best ranked systems according to the 2008 test of the University of Applied Sciences Berlin (HTW) [5]:

Copyscape[2], first and third place (premium and free version), a plagiarism checker of Indigo Stream Technologies Ltd. Given a text it searches the Internet for possible plagiarism of this text using the document's words in the given order.

Plagiarism Detector[3] (scored second place) by SkyLine, Inc. uses non-overlapping n-grams with a configurable spacing between them to find online plagiarisms of a given text in various possible input formats.

Urkund[4] (scored fourth place), by PrioInfo AB targets to check papers written by students for possible plagiarism and searches the Internet (with known paper mills), an own corpus of scientific publications, and papers checked for plagiarism before.

WCopyfind[5] (This system has no HTW ranking but was marked as "good" in the 2008 test.) is an n-gram based plagiarism checker of the University of Virginia, Charlottesville [6]. It targets students' papers, searching a corpus which has to be compiled by the user. Since it is open source software, this tool was used as a comparison for our SemDupl system.

[2] http://www.copyscape.com/
[3] http://plagiarism-detector.com/
[4] http://www.urkund.de/
[5] http://plagiarism.phys.virginia.edu/Wsoftware.html

3. The SemDupl Corpus

The corpus used in the learning process and for evaluation purposes (see Section 8) is composed of the following manually annotated subcorpora:

- RSS news (semdupl-rss): News feed articles of different German media consisting of 99 texts annotated with 113 duplicate pairs[6].

- Prose (semdupl-prose): Short stories by Edgar Allen Poe translated to German by different translators (split in 136 parts of about 600 words each) with 68 duplicate pairs.

- Internet (semdupl-google): 100 texts collected from Google (the 10 top texts of the 10 fastest growing search terms in 2008), containing 42 duplicate pairs.

- Plagiarism (htw): Weber-Wulff's collection of plagiarisms (slightly extended), annotated as 77 texts with 39 duplicate pairs.

- Minimal test units (semdupl-units): A collection of (mostly) minimal text pairs. These were written to test the deep duplicate detector for single paraphrase phenomena (see Section 5.1). This subcorpus contains 82 texts with 70 duplicate pairs. Note that this subcorpus was not used in the evaluation.

4. Shallow Approaches

SemDupl uses two shallow approaches as filters on large corpora and/or as a robust fall-back strategy (if deep parsing fails). These two approaches are discussed in the following subsections.

4.1 CErkenner (CE)

To detect whether a text is the duplicate of another text, CE uses a set of 39 features derived from the surface structure of the texts which include the following:

- Word sets: The words of the compared texts represented as sets, with elements being the text's words as they are given, without stop words, in stemmed form or united with their synonyms.

- Length of words and sentences: Weber-Wulff [7] states that plagiarized texts often share the same style of writing. Since a writers style includes the average length of words and sentences (per paragraph), these two values are used as features in the process.

- N-grams: Word n-grams are sequences of *n* words from the texts. In CErkenner the different types of word n-grams used are simple n-grams, alliterations (n-grams where all words start with the same letter), phonetic alliterations (alliterations with the words sharing the same initial phoneme) and k-skip-n-grams, i.e., n-grams where up to *k* words are left out (“skipped”) between the elements of the n-grams [8].

CE combines these features using machine-learning techniques and was trained using the SemDupl corpus.

4.2 ShallowChecker (SC)

Tests indicated that the shallow approach of CE achieves good results regarding precision and accuracy, but due to its time complexity it is rather unsuited for large corpora. So another shallow approach was devised using only features which can be calculated efficiently.

In the preprocessing phase, the ShallowChecker (SC) searches the given texts for misspelled words and words with a frequency class above a given threshold and compiles all n-grams with lengths from 3 to 7. These values are used as indices whereas the text's id (e.g., file name) is used as value. This generates a database with a list of texts for each value (with a table for each feature).

In the detection phase, all rows *r* containing a given text id are searched inside the tables. For each *other* affected text id found inside these rows the ratio between the number of rows in *r* containing the affected text id and the total number of rows *r* is calculated for each table (and therefore feature). These scores are combined linearly and normalized, resulting in a combined score for each text pair. A text is regarded as a duplicate if the score is greater than a given threshold.

[6] The reported numbers include only non-trivial duplicates, i.e., the pairs made of the same document and symmetric variations are excluded.

4.3 Comparison of Shallow Approaches

CErkenner works on texts without any major preprocessing: it is ready to instantly check an arbitrary pair of texts without *any* preprocessing steps as an "out-of-the-box" duplicate detector. Its capability to learn the "definition" of duplicates on an annotated corpus leads to a detection which has a lower chance of failing because of bad user-set thresholds. Its downside is that it has to inspect every possible text pair in order to detect all duplicates in a given corpus, resulting in quadratic time complexity, so it should be used on small or filtered corpora.

ShallowChecker, on the other hand, uses preprocessing resulting in a lower time complexity while detecting, but only some of the possible features can be used as index-values and the thresholds, which are defined by the user, may, if not set well, become a source of errors.

5. Linguistic Phenomena Relevant for Semantic Duplicates

5.1 Types of Paraphrases for Semantic Duplicates

Many problems exist for standard surface oriented comparisons for duplicate detection; here are some examples:

- different word forms due to inflection
- different orthography (e.g., new and old orthography in German)
- abbreviations/acronyms and expanded forms
- different hyphenation of compounds
- different word order (especially relevant in German)
- discontinuous word forms (e.g., German verbs with separable prefix)
- different voices (active or passive in German)
- nominalization of situations, e.g., *"discussion"* vs. *"to discuss"*
- information distribution across sentences
- synonyms: partially solved by HaGenLex plus GermaNet (relation SYNO); for compounds, many synonyms can be inferred from synonyms of parts
- hyponyms: e.g., *"dentist"* vs. *"physician"*, solved by lexico-semantic relations
- compounds vs. analytical expressions like complex NPs and clauses: e.g., *Finanzierungslücke*/*"finance gap"* vs. *Lücke bei der Finanzierung/"gap in financing"*
- idioms: An idiom lexicon of 250 idioms based on verbs is employed.
- support verb constructions (SVCs), *"to utter an objection"* vs. *"to object"*. In SempDupl, this is achieved by MultiNet rules (derived from a SVC lexicon) that are applied during query expansion.
- coreferences (different expressions referring to the same entity): solved by the coreference module.
- entailments (especially entailments of verbs), e.g., *"to buy"* vs. *"to sell"*; covered in part by entailments from HaGenLex and entailments derived from knowledge bases like GermaNet and manual translations of XWordNet.

Most of the above paraphrase problems are tackled by the parser and its modules; limitations have been mentioned above.

A nice example from our semdupl-prose subcorpus shows that these phenomena combine quite often:

… sagte Dupin, während er seinem Besuch eine Pfeife reichte und einen bequemen Sessel hinschob. / *"Dupin … said, while he passed his visitor a pipe and moved a comfortable chair to him"* vs. *… antwortete Dupin, während er den Gast mit einer Pfeife versorgte und einen bequemen Sessel heranschob./"Dupin … replied, while he provided his guest with a pipe and moved a comfortable chair up to him".* The two sentences can only be reliably linked as nearly synonymous if four links can be constructed:

- *hinschieben* and *heranschieben* (different forms of *"to move"*) can be linked as cohyponyms;
- *reichen/"to pass"* can be related to *versorgen/ "to provide"* via a verb entailment represented at "*versorgen*" (in addition, hypernyms for "reichen" must be available);
- *antworten*/*"to reply"* as a troponym of *sagen*/ *"to say"*; and
- *Gast*/*"guest"* and *Besuch*/*"visit(or)"* as synonyms.

5.2 Restrictive Contexts and Other Precision Problems for Semantic Duplicates

Precision is less of a problem (compared to recall) for a deep approach; nevertheless some phenomena must be controlled to preserve precision:

- incorrect phrases: solved by parsing sentences.
- incorrectly selected reading (wrong reading of ambiguous word or constituent): this is partially solved by the disambiguation modules in the parser.
- negation; constituent negation (compatibility test for the **fact** layer feature in MultiNet suffices); sentence negation, similarly.
- other modalities. Incompatible modalities are tested in the SN representations. Similarly, hypothetical situations must be excluded from matching real situations. Other examples of modality come from epistemic modals like *glauben*/*"to believe"*.

6. Knowledge Acquisition for Deep Duplicate Detectors

The deep duplicate detector can only be as good as the underlying knowledge bases. Therefore, the SemDupl project tries:

- to consolidate our existing knowledge sources,
- to automatically (or semi-automatically) derive new knowledge bases, and
- to validate these new knowledge bases.

6.1 Hypernym Acquisition

A type of near-duplicates that is both quite easy to create and to detect is a pair of sentences being almost identical except that certain words (or concepts on a semantic level) of the original sentence are replaced by hypernyms. This is a method often used while trying to obfuscate plagiarism.

For example, *"His father buys a new laptop."* implies *"His father buys a new computer."* In the second sentence, *"laptop"* is replaced by one of its hypernyms, *"computer"*. Thus, a large collection of hypernyms is quite vital for near-duplicate recognition.

Since Wikipedia is often used as a source for plagiarisms or duplicates, hypernyms are extracted from Wikipedia using a pattern-based approach [9], differentiating between shallow and deep patterns.

Both types of patterns consist of a premise and a conclusion part where the conclusion part is of the form (*a* SUB0 *b*) which specifies that, if the premise holds, a hypernymy relationship between the concepts which are assigned to the variables *b* and *a* holds. The assignments for both variables are determined by matching the premise part to a linguistic structure which is created by analyzing the associated sentence. The premise of a shallow pattern is given just by a regular expression which is tried to be matched with the token list of a sentence. In contrast, the premise of a deep pattern is given as an SN. This SN is tried to be matched to the SN of a sentence by a graph pattern matcher (or an automated theorem prover if axioms are to be employed).

An example pattern is given in Equation 1 and Figure 2.

$$(a1\ \mathrm{SUB0}\ a2) \leftarrow\ [c{=}(^*\mathrm{DIFF}\ d\ e)] \wedge (d\ \mathrm{PRED}\ a2) \wedge (e\ \mathrm{PRED}\ a1) \wedge (e\ \mathrm{SUBM}\ d) \qquad (1)$$

Figure 1 illustrates the application of the deep pattern that is displayed in Figure 2. This pattern can be employed to extract the hyponymy relation (*cello.1.1* SUB0 string_instrument.*1.1*) from the sentence: *Er verkauft alle gebräuchlichen Streichinstrumente außer Celli./"He sells all common string instruments except celli."* Note that we consider *instance of*-relations as a special kind of hyponymy as well and such relations were also extracted by our algorithm.

6.2 Deep Semantic vs. Syntactic and Shallow Patterns

On the one hand, a shallow pattern has the advantage that it is also applicable if the parse fails. It only relies on the fact that the tokenization is successful. On the other hand, deep patterns are still applicable if there are additional constituents or subclauses between the hyponym and the hypernym, which usually cannot be handled well by shallow patterns.

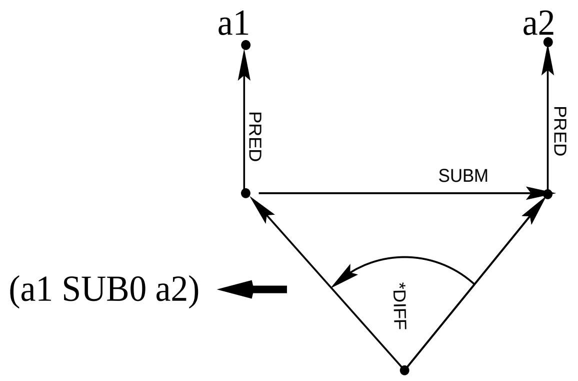

Figure 2. Deep pattern for hypernymy extraction where the premise is given as an SN.

Another advantage of deep patterns is illustrated by the following sentences:

- *Er verkauft alle gebräuchlichen Streichinstrumente außer Celli./“He sells all common string instruments except celli.”*
- *Er verkauft alle gebräuchlichen Streichinstrumente bis auf Celli./“He sells all common string instruments aside from celli.”*
- *Er verkauft alle gebräuchlichen Streichinstrumente ausgenommen Celli./“He sells all common string instruments excluding celli.”*

All three sentences have different dependency trees (tested by applying the Stanford Dependency Parser [10] on the English translations). However the SN representations of all three sentences are identical (depicted in Figure 1), i.e., the pattern given in Figure 2 can be applied to extract the hyponymy relation *(cello.1.1* SUB0 *string_instrument.1.1)* from all three sentences while three different syntactic patterns would have to be designed if the same relation was to be extracted from the dependency parse. The same fact holds if surface representations are used. Thus, the use of a deep semantic representation reduces the amount of required patterns in comparison to a surface or dependency based representation.

A further advantage of the deep semantic approach consists in the fact that person names are already identified by the parser which simplifies the extraction of hyponyms (instance-of relations) relating to persons.

Finally, the deep approach allows the usage of logical axioms, which can make the patterns more generally applicable.

6.3 Extraction of Entailments

Next to semantic relations, a collection of entailments can be useful for plagiarism or duplicate detection. An entailment is a relationship between two expressions which holds, if the truth of the first expressions (the base expression/text) implies the truth of the other (the hypothesis). We basically follow the approach of Ravichandran and Hovy which identifies entailments by collecting texts with identical noun phrases by using the assumption that such texts often contain similar contents [11].

Consider for example the noun phrases: “John McEnroe”, “Björn Borg”, “Wimbledon”, “1980”. Sentences containing such sentences could be:

- “***John McEnroe*** *lost against* ***Björn Borg*** *in the final of* ***Wimbledon 1980****.*” or
- “***Björn Borg*** *beat* ***John McEnroe*** *in the final of* ***Wimbledon 1980****.”*

Thus, if a lot of such texts are examined, the entailments “X beat Y” → “Y lost against X” and
“Y lost against X”→ “X beat Y” can eventually be learned. The entailment extraction is done in the following steps:

- SNs are created for all texts which contain given noun phrases. The texts were extracted by web search engine queries.
- Nominal phrases which are employed for the search are replaced by variables.
- Frequently occurring substructures S in these SN are learned by following the Minimum Description Length Principle [12].

Entailments are created by building the Cartesian product over S: S×S, i.e., the first component of a pair $s \in S \times S$ represents the base expression, the second the hypothesis.

7. Deep Duplicate Detector

To handle linguistic phenomena adequately, i.e., identify paraphrase phenomena discussed in Section 5.1 and to not get disturbed by non-paraphrase phenomena discussed in Section 5.2, a deep semantic approach to duplicate detection has been developed. It integrates existing tools for producing semantic representations for texts: the WOCADI parser and the CORUDIS coreference resolver [13]. In an indexing phase, all texts in the base corpus are transformed into semantic representations by WOCADI and CORUDIS.

In the detection step, the duplicate candidate (a text) is analyzed in the same way as the texts of the base corpus. For each sentence in the candidate, a semantic search query is sent to a retrieval system that contains all the semantic representations for the base corpus. Matches are collected and, after all sentences of the candidate have been investigated, scores are calculated from the results for the text sentences. The average overlap score over all candidate sentences is a good score. The individual overlap score is calculated by the retrieval system, based on semantic distances of related concepts and the distance between the left-hand side and right-hand side of inference rules.

8. Evaluation

The three individual detectors as well as the combined system have been evaluated on the SemDupl corpus, which is annotated for duplicates. For each text pair and each approach, a set of feature values is generated where high values indicate the texts being duplicates. These values are combined by the support vector machine classifier WLSVM [14], which is based on LIBSVM [15]. For training this classifier, the text pairs of our corpus were used (in 10-fold cross-validation). The confusion matrices calculated for shallow and deep approaches are shown in Tables 1-4.

	WCopyfind		SC	
	PD	PND	PD	PND
D	39	215	97	157
ND	8	21645	16	21637

Table 1: Confusion matrices for the shallow approaches WCopyfind and SC, D=Duplicate, ND=No Duplicate, PD=Predicted Duplicate, PND=Predicted Non-Duplicate.

	CE		SC+CE	
	PD	PND	PD	PND
D	200	54	201	53
ND	14	21639	13	21640

Table 2: Confusion matrices for the shallow approaches CE and SC+CE.

	PD	PND
D	42	212
ND	5	21648

Table 3: Confusion matrices for the deep approach DC.

	DC+SC		DC+SC+CE	
	PD	PND	PD	PND
D	106	148	202	52
ND	16	21637	11	21642

Table 4: Confusion matrices for deep+shallow approaches.

We also determined the run times of our duplicate checkers. For all text comparisons in our SemDupl corpus, the ShallowChecker required 0.3 minutes, the CErkenner 158 minutes, and the DeepChecker 240 minutes. This test was run on a computer with a Core i7 920 (2.67 GHz) processor and 6GiB RAM.

9. Interpretation and Conclusion

In order to compare the results of the combined system with plagiarism detection software, *WCopyfind* was evaluated on our text corpus, see Table 1 and 5 for details. Tables 5 and 6 show the results of our approach. It can be seen that each approach of our system generates significantly better results in terms of F-measure, precision, and recall than WCopyfind (determined with confidence intervals of level 99%).

Measure	WCopy-find	SC	CE	SC+CE
F-measure	0.259	0.529	0.855	0.859
Precision	0.830	0.858	0.935	0.939
Recall	0.154	0.382	0.787	0.791
Accuracy	0.990	0.992	0.997	0.997

Table 5: F-measure, precision, recall, and accuracy for shallow approaches.

Measure	DC	DC+SC	DC+SC+CE
F-measure	0.279	0.564	0.865
Precision	0.894	0.869	0.948
Recall	0.165	0.417	0.795
Accuracy	0.990	0.993	0.997

Table 6: F-measure, precision, recall, and accuracy for deep+shallow approaches.

Similarly (at 95% confidence), the best combined shallow+deep approach outperforms the best shallow approach. Furthermore, F-measure, precision, and recall of the combined shallow+deep system are considerably higher than the associated values of the combination of the two shallow systems. Note that the DC approach can show its full potential only in more professionally constructed duplicates; for example, on the semdupl-units subcorpus (which was excluded from the evaluation because it was constructed), it performs much better than any shallow system. In future work, we want to improve the coverage of the deep approach by further extending its knowledge bases by (semi-) automatic means.

Acknowledgments

Hereby we thank all members of our department who supported us. This work was funded by the DFG project *Semantische Duplikatserkennung mithilfe von Textual Entailment* (HE 2847/11-1).

References

[1] Stefan Schaltenbrand, Alles gestohlen? Vom Plagiat zur Wiederholung (*Everything Stolen? From Plagiarism to Repetition*), Frieling & Partner, Berlin, Germany, 1994

[2] Christian Eichhorn, Automatische Duplikats-erkennung – Ähnliche Texte entdecken und erkennen (*Automatic Duplicate Detection*), Der Andere Verlag, Tönning, Germany, 2010

[3] Gisela Hüttinger, Software zur Plagiatserkennung im Test – die Systeme haben sich deutlich gebessert (*Test of Plagiarism Detection Software*), Hochschule für Technik und Wirtschaft Berlin (HTW), Berlin, Germany, online at: http://www.htw-berlin.de/Aktuelles/Pressemitteilungen/2008/index.html,2008

[4] Hermann Helbig, Knowledge Representation and the Semantics of Natural Language, Springer, Heidelberg, Germany, 2006

[5] Debora Weber-Wulff, Softwaretest 2008, online at: http://plagiat.htw-berlin.de/software, 2008

[6] Enrique V. Balaguer, "Putting Ourselves in SME's Shoes: Automatic Detection of Plagiarism by the WCopyfind Tool", Proceedings of PAN Workshop and Competition, Valencia, Spain, pp. 34-35, 2009

[7] Debora Weber-Wulff, "Der große Online Schwindel (*The Big Deception*)", Spiegel Online, Hamburg, Germany, online at: http://www.spiegel.de/unispiegel/studium/0,1518,221507,00.html, 2002

[8] David Guthrie, Ben Allison, Wei Liu, Louise Guthrie and Yorick Wilks, "A Closer Look at Skip-Gram Modelling", Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC), Geneva, Switzerland, pp. 1222-1225, 2006

[9] Marti Hearst, "Automatic Acquisition of Hyponyms From Large Text Corpora", Proceedings of the 14th International Conference on Computational Linguistics (COLING), Nantes, France, pp. 539-545, 1992

[10] Marie-Catherine de Marneffe and Christopher D. Manning, "Stanford Typed Dependencies Manual", online at: http://nlp.stanford.edu/software/dependencies_manual.pdf, Manual, 2008

[11] Deepak Ravichandran and Eduard Hovy, "Learning Text Patterns for a Question Answering System", Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, Pennsylvania, pp. 41-47, 2002

[12] Diane J. Cook and Lawrence B. Holder, "Substructure Discovery Using Minimum Description Length and Background Knowledge", Journal of Artificial Intelligence, pp. 231-255, 1(1), 1994

[13] Sven Hartrumpf, Hybrid Disambiguation in Natural Language Analysis, PHD thesis, Der Andere Verlag, Osnabrück, Germany, 2003

[14] Yasser El-Manzalawy and Vasant Honavar, WLSVM: Integrating LibSVM into Weka Environment, Manual, online at: http://www.cs.iastate.edu/~yasser/wlsvm, 2005

[15] Chich-Chung Chang and Chih-Jen Lin, "LIBSVM: A Library for Support Vector Machines", Manual, online at: http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001