Semantic Duplicate Identification with Parsing and Machine Learning

Sven Hartrumpf[1], Tim vor der Brück[1], Christian Eichhorn[2]

[1] Intelligent Information and Communication Systems (IICS)
FernUniversität in Hagen, 58084 Hagen, Germany
{sven.hartrumpf,tim.vorderbrueck}@fernuni-hagen.de
[2] Lehrstuhl Informatik 1
Technische Universität Dortmund, 44227 Dortmund, Germany
christian.eichhorn@tu-dortmund.de

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 index. In order to detect many kinds of paraphrases the semantic networks of a candidate text are varied by applying inferences: lexico-semantic relations, relation axioms, and meaning postulates. Important phenomena occurring in difficult duplicates are discussed. The deep approach profits from background knowledge, whose acquisition from corpora is explained briefly. The deep duplicate recognizer is combined with two shallow duplicate recognizers in order to guarantee a 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.

1 Introduction

With the growth of the web, the number of available texts has increased rapidly.[3] The number of duplicates increased with similar speed, deliberately by generating plagiarisms or unwittingly by presenting information already given by other users or services.

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

[3] We would like to thank all members of our departments who supported us. This work was funded by the DFG project *Semantische Duplikatserkennung mithilfe von Textual Entailment* (HE 2847/11-1).

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 [1, Sect. 3.2]. Even if the capability of these approaches has increased, they are still capable of detecting only three quarters of the tested plagiarisms [2].

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[4] 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.

2 State of the Art

As stated, 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 (FHTW) [4].

Copyscape [5], 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 [6] 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 [7] 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 [8] is an n-gram based plagiarism checker of the University of Virginia, Charlottesville [5]. It targets student's 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.

[4] The formalism is MultiNet, Multilayered Extended Semantic Networks, [3].
[5] http://www.copyscape.com/, first and third place (premium and free version)
[6] http://plagiarism-detector.com/, scored second place
[7] http://www.urkund.de/, scored fourth place
[8] http://plagiarism.phys.virginia.edu/Wsoftware.html, marked as "good"

3 The SemDupl Corpus

The corpus used in the learning process of the shallow duplicate checker CErkenner (see Sect. 4.1) and for evaluation purposes (see Sect. 8) contains 287,044 words in 13,622 sentences. It 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[9].

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

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).

4.1 CErkenner (CE)

To detect whether a text is the duplicate of another text, CE uses 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.

Typos Weber-Wulff [6] shows that a promising strategy in searching for plagiarism is to compare the spelling mistakes in different texts, since if a text is plagiarized, its typos are often copied, too.

Length of words and sentences Weber-Wulff [6] 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 (n-grams where up to k words are left out ("skipped") between the elements of the n-grams [7]).

CE combines these features using machine-learning techniques and was trained using the SemDupl corpus (see Sect. 3).

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

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

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 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 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:

1. different word forms due to inflection
2. different orthography (e.g. new and old orthography in German).
3. abbreviations/acronyms and expanded forms
4. different hyphenation of compounds
5. different word order (especially relevant in German)
6. discontinuous word forms (e.g. German verbs with separable prefix)

7. different voices (active or passive in German)
8. nominalization of situations, e.g. *discussion* vs. *to discuss*; supported by around 3,000 verb-noun links in the lexicon (HaGenLex, [8])
9. information distribution across sentences
10. synonyms: partially solved by HaGenLex plus GermaNet (relation SYNO)
11. hyponyms: e.g. *dentist* vs. *physician* solved by lexico-semantic relations.
12. compounds vs. analytical expressions like complex NPs and clauses: e.g. *finance gap* vs. *gap in financing* (ca. 500,000 compound analyses available)
13. idioms: An idiom lexicon of 250 idioms based on verbs is employed.
14. support verb constructions (SVCs), e.g. *to utter an objection* vs. *to object*. In SemDupl, this achieved by around 500 MultiNet rules (derived from a SVC lexicon) that are applied during query expansion.
15. coreferences (different expressions referring to the same entity): solved by the coreference module.
16. entailments 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 (several thousand rules).

Most of the above paraphrase problems are tackled by the WOCADI parser (see Sect. 7) 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* vs. The two sentences can only be reliably linked as nearly synonymous if four links can be constructed:

1. *hinschieben* and *heranschieben* can be linked as cohyponyms;
2. *reichen/to pass* can be related to *versorgen/to provide* via verb entailment represented at *versorgen* and a troponym for *reichen*;
3. *antworten/to reply* as a troponym of *sagen/to say*; and
4. *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 for a deep approach; nevertheless some phenomena must be controlled to preserve precision even in a deep approach:

1. incorrect phrases: solved by parsing sentences
2. incorrectly selected reading (wrong reading of ambiguous word or constituent)
3. negation; constituent negation (compatibility test for the FACT layer feature in MultiNet suffices); sentence negation, similarly.
4. other modalities. Incompatible modalities are tested in the semantic networks. Similarly, hypothetical situations must be excluded from matching real situations. Other examples of modality come from epistemic modals like *glauben/to believe*.

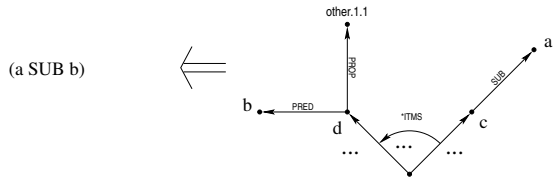

Fig. 1. Deep pattern for hypernymy extraction (premise as a semantic network).

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 (1) consolidate our existing knowledge sources, (2) automatically (or semi-automatically) derive new knowledge bases, and (3) validate these new knowledges 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 source for plagiarisms or duplicates, hypernyms and holonyms are extracted from Wikipedia using a pattern-based approach, differentiating between shallow and deep patterns.

Both types of patterns consist of a conclusion part of the form (a SUB b) which specifies that, if the premise holds, a hypernymy relationship between the concepts which are assigned to the variables a and b 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 a semantic network graph. This graph is tried to be matched to the semantic network 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 Fig. 1.

$$(a \text{ SUB } b) \leftarrow \underset{*_{\text{ITMS}}}{follows}(c, d) \wedge (d \text{ PRED } b) \wedge (d \text{ PROP} other.1.1) \wedge (c \text{ SUB } a) \quad (1)$$

$follows_{*_{\text{ITMS}}}(c, d)$ denotes the fact that c precedes d in the argument list of the function *ITMS. This pattern can be employed to extract the hypernymy relation ($cello.1.1$ SUB $instrument.1.1$) from the sentence: *The old man owns a cello and other instruments.* Note that we consider *instance of* relations as a special kind of hypernymy as well and such relations were also extracted by our algorithm.

6.2 Deep vs. 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 between hyponym and hypernym, where shallow patterns often fail.

Another advantage of deep patterns is illustrated by the following sentence: *In any case, not all incidents from the Bermuda Triangle or from other world areas are fully explained.* From this sentence, a hypernymy pair cannot be extracted by the Hearst pattern *X and other Y* [9]. The application of this pattern fails due to the word *aus/from* which cannot be matched. To extract this relation by means of shallow patterns an additional pattern would have to be introduced. This could also be the case if syntactic patterns were used instead since the coordination of *Bermuda Triangle* and *world areas* is not represented in the syntactic constituency tree but only on a semantic level[10]. Thus, the same deep pattern can be used for the hypernymy extraction in this sentence as for the following phrase: *the Bermuda Triangle or other world areas.*

Furthermore, different syntactic or surface representations are frequently mapped to the same semantic network, e.g.:

1. *He owns a cello, a violin and other instruments.*
2. *He owns a violin, a cello as well as other instruments.*

Thus, the hyponymy relationships that cellos and violins are instruments can be extracted by the application of the same deep pattern. However, to extract the same information by the application of shallow patterns two different patterns have to be defined. Finally, the deep approach allows the usage of logical axioms, which can make, by using inferences, the patterns more generally applicable.

7 Deep Duplicate Detector (DC)

To handle linguistic phenomena adequately, i.e., identify paraphrase phenomena discussed in Sect. 5.1 and to not get disturbed by non-paraphrase phenomena discussed in Sect. 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 [10]. 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 (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,

[10] Note that some dependency parsers normalize some syntactic variations too.

Table 1. Confusion matrices for shallow and deep approaches. D=Duplicate, ND=No Duplicate.

	SC		CE		SC+CE	
	D	ND	D	ND	D	ND
D	97	157	200	54	201	53
ND	16	21637	14	21639	13	21640

	DC		DC+SC		DC+SC+CE	
	D	ND	D	ND	D	ND
D	42	212	106	148	202	52
ND	5	21648	16	21637	11	21642

Table 2. F-measure, precision, recall, and accuracy.

	Shallow approaches				Deep approaches		
Measure	WCopyFind	SC	CE	SC+CE	DC	DC+SC	DC+SC+CE
F-measure	0.259	0.529	0.855	0.859	0.279	0.564	0.865
Precision	0.830	0.858	0.935	0.939	0.894	0.869	0.948
Recall	0.154	0.382	0.787	0.791	0.165	0.417	0.795
Accuracy	0.990	0.992	0.997	0.997	0.990	0.993	0.997

based on distances of related concepts and the distance between the left-hand side and right-hand side of inference rules. The average detection time (for parsing of the candidate text and querying the SemDupl corpus) was around 30 seconds on a PC with one CPU core.

8 Evaluation

The three individual detectors as well as the combined system have been evaluated on the SemDupl corpus (see Sect. 3), which is annotated for duplicates. For each text pair and each approach, a set of features values is generated where high values indicate the texts being duplicates. These values are combined by the support vector machine classifier WLSVM [11], which is based on libsvm [12]. For training this classifier, the text pairs of our corpus were used (in ten-fold cross-validation). The confusion matrices calculated for shallow and deep approaches are shown in Table 1.

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, too. Table 2 shows the results of our approaches. Precision is the relative frequency that a

system-reported duplicate pair is a duplicate pair in the gold standard annotation. In contrast, accuracy looks at all decisions for given document pairs: the relative frequency that a document pair is correctly classified as duplicate or non-duplicate. Accuracy values are very high because the class of duplicates is tiny compared to the class of non-duplicates.

Except for the single DC approach, each approach of our system generates significantly better results in terms of precision, recall, and F-measure than *WCopyFind* (significance level of 1%). Note that the DC approach can show its full potential only in more professionally constructed duplicates. Furthermore, F-measure, precision, and recall of the combined shallow+deep system are significantly higher (significance level of 5%) than for the combination of the two shallow systems. We want to improve the coverage of the deep approach by further extending its knowledge bases by (semi-) automatic means.

References

1. Eichhorn, C.: Automatische Duplikatserkennung (*Automatic Duplicate Detection*). Der Andere Verlag, Tönning, Germany (2010)
2. Hüttinger, G.: Software zur Plagiatserkennung im Test - die Systeme haben sich deutlich gebessert (*Test of plagiarism detection software*) (November 2008) http://www.htw-berlin.de/Aktuelles/Pressemitteilungen/2008/index.html.
3. Helbig, H.: Knowledge Representation and the Semantics of Natural Language. Springer, Berlin, Germany (2006)
4. Weber-Wulff, D.: Softwaretest 2008 (2009) online at http://plagiat.fhtw-berlin.de/software.
5. Balaguer, E.V.: Putting ourselves in SME's shoes: Automatic detection of plagiarism by the WCopyFind tool. In: Proceedings of PAN Workshop and Competition, Valencia, Spain (2009)
6. Weber-Wulff, D.: Der große Online-Schwindel (*The big online deception*). Spiegel Online (2002)
7. Guthrie, D., Allison, B., Liu, W., Guthrie, L., Wilks, Y.: A closer look at skip-gram modelling. In: Proceedings of the Fifth international Conference on Language Resources and Evaluation (LREC), Geneva, Switzerland (2006) 1222–1225
8. Hartrumpf, S., Helbig, H., Osswald, R.: The semantically based computer lexicon HaGenLex – Structure and technological environment. Traitement automatique des langues 44(2) (2003) 81–105
9. Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of COLING, Nantes, France (1992)
10. Hartrumpf, S.: Hybrid Disambiguation in Natural Language Analysis. Der Andere Verlag, Osnabrück, Germany (2003)
11. EL-Manzalawy, Y., Honavar, V.: WLSVM: Integrating LibSVM into Weka Environment. (2005) Software available at http://www.cs.iastate.edu/~yasser/wlsvm.
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.