PG588: SciencePlag

Ole Bergenholtz Jonas Holtorf Marc Jasper Raoul Lefmann
Kevin Nikiel Lutz Oettershagen Philip Rehm

Lehrstuhl 11, Fakultät für Informatik, Technische Universität Dortmund

Plagiatserkennung

Motivation

- Akademische Plagiate (Guttenberg, Schavan, von der Leyen) [3]
- Finden von Plagiaten schwierig und langwierig
- Aktuelle Software nur teilweise nützlich bis nutzlos
- Erleichtern der Plagiatserkennung durch Software

Zielsetzung

- Software zur effizienten Plagiatserkennung
- Methoden aus Stringology und Information Retrieval
- Einlesen von Textdokumenten und Abschlussarbeiten

Methodik

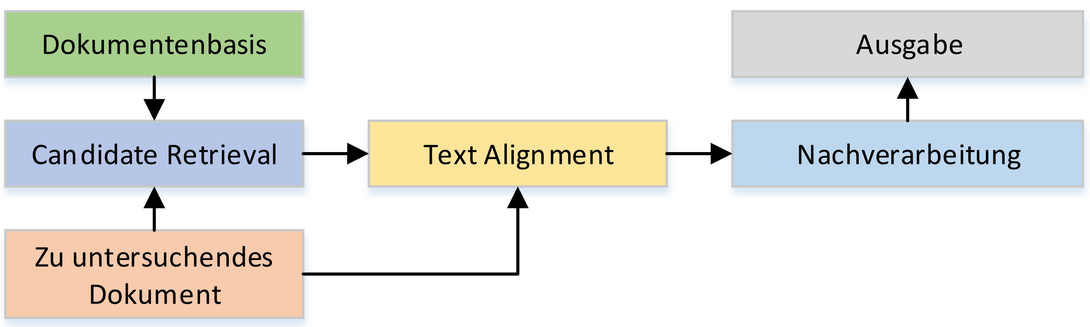

Abbildung: Vorgehensweise der Software

Schritte der Software

- Vorverarbeitung der Dokumente
- Aufbau einer Dokumentenbasis
- Kandidatendokumente mit Methoden des Information Retrieval suchen
- Ähnliche Stellen mit Methoden der Stringology finden
- Nachverarbeitung und Ausgabe möglicherweise plagiierter Stellen

Vorverarbeitung

- *Tokenization*: Aufteilen nach Satzzeichen
- *Stemming*: Reduzierung von Wörtern auf den Stamm
- *Stop Word Removal*: Wörter mit wenig Informationsgehalt entfernen
- *Normalisierung*: Entfernung von Sonderzeichen
- *Integer-Alphabet*: Wörter werden auf Zahlen abgebildet

Candidate Retrieval

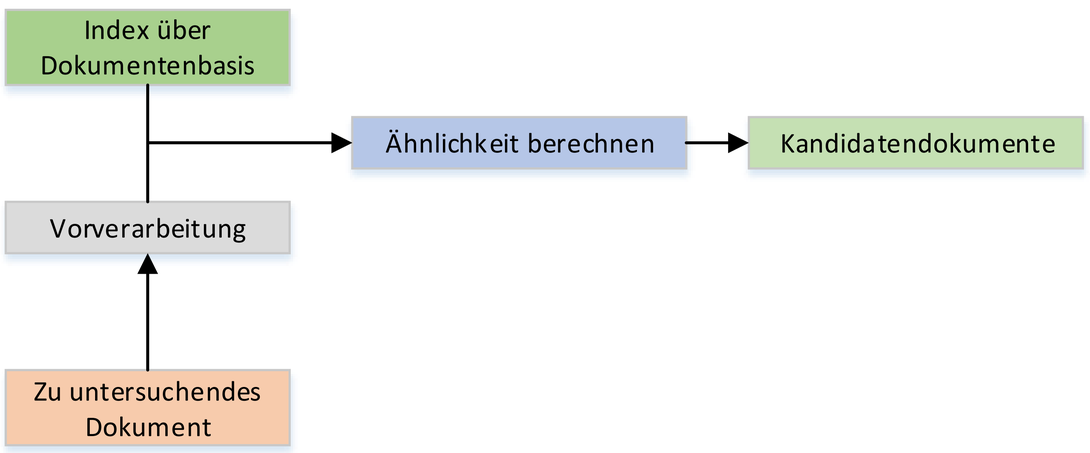

Abbildung: Vorgehensweise bei Candidate Retrieval

- Eingabe ist eine Dokumentenbasis und ein zu untersuchendes Dokument (Anfrage)
- Vollständige Suche in der Dokumentenbasis nach Dokumenten, die ähnlich zur Anfrage sind
- Kandidatendokumente sind Teilmenge der Dokumentenbasis

Ähnlichkeit von Dokumenten

- Repräsentation der Dokumente als *tf-idf-Vektor* [2]
  - *tf*: Term Frequency - Häufigkeit des Wortes im Dokument
  - *idf*: Inverse Document Frequency - Größe der Dokumentenbasis geteilt durch Anzahl der Dokumente, die dieses Wort enthalten
  - *tf-idf-Vektor*: Vektor über allen Wörtern, die im Dokument vorkommen. Die i-te Zeile repräsentiert den $tf \cdot idf$ Wert des Wortes i im Dokument d.
- Ähnlichkeit zweier Dokument-Vektoren d und d' mit Kosinus-Ähnlichkeit berechnen: $\text{sim}(d, d') := \cos(\angle(d, d'))$
- Ausgabe von Dokumenten, deren Ähnlichkeit einen Schwellwert überschreitet

Text Alignment mit Suffixarrays

```
k  = 1 6 7 9 7 4 3 5 4 9 3
t1 = 3 4 5 2 7 1 6 7 9 1 6
t2 = 9 3 5 1 2 3 5 4 7 8 1
T  = 1 6 7 9 7 4 3 5 4 9 3 3 4 5 2 7 1 6 7 9 1 6 9 3 5 1 2 3 5 4 7 8 1
```

i	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	...
SA	33	26	17	01	21	27	15	11	12	24	28	07	06	13	30	09	25	
LCP	-	1	1	4	2	0	1	0	1	1	2	3	0	1	1	1	0	

Abbildung: Beispiel eines Suffixarrays (SA)

- k : Text des verdächtigen Dokuments
- t_1, t_2 : Text der Kandidatendokumente
- Grau unterlegt: Enhanced Suffix Array [1]
  - SA: Suffixarray: Lexikographisch sortierte Wortketten
  - LCP: Longest Common Prefix Array
- Rot markiert: Fundstelle

Nachbearbeitung

- *Merge*: Vereinigung sich überlappender Fundstellen
- *Delete*: Häufig vorkommende Formulierungen entfernen
- *Dominate*: Gleiche Fundstellen in verschiedenen Kandidatendokumenten beseitigen

Ausgabe

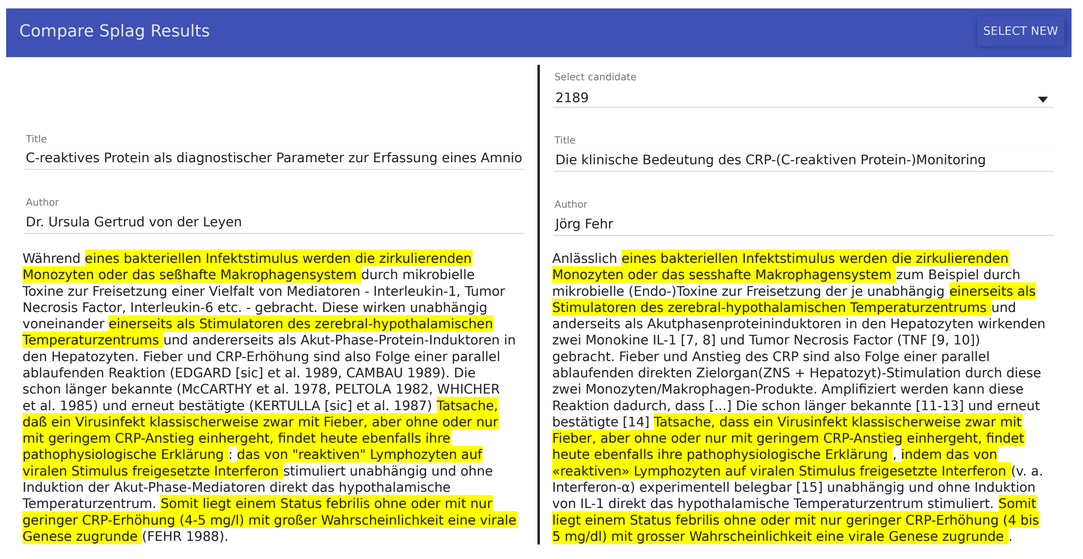

Abbildung: Beispielausgabe des Programms

Referenzen

[1] MANBER, Udi ; MYERS, Gene:
Suffix arrays: a new method for on-line string searches.
In: *SIAM Journal on Computing* 22 (1993), Nr. 5, S. 935–948

[2] SPARCK JONES, Karen:
A statistical interpretation of term specificity and its application in retrieval.
In: *Journal of documentation* 28 (1972), Nr. 1, S. 11–21

[3] WEBER-WULFF, Debora:
False feathers: a perspective on academic plagiarism.
Springer Science & Business, 2014