Seminar "Automatisierte, interaktive und kollaborative Analyse von Textähnlichkeiten"

Infos

- **Ort:** CIP-Pool PHY 1.0.02
- **Zeit:** Donnerstags, 14-16 Uhr (+ optionale freie Projektzeit von 16-18 Uhr)
- Informationen im LSF

Aktuell

Bis 31. Mai 2012:

Lest Tutorials zu Git, probiert sie aus

- http://coding.smashingmagazine.com/2011/07/26/modern-version-control-with-git-series/
- http://learn.github.com/p/intro.html
- http://en.wikibooks.org/wiki/Git/Introduction
- http://tomb.io/posts/branches-everyone-should-use-them/

Misc:

- Maike Krüger, Elisabeth Salomon, Philipp Jean-Jacques am 31. Mai nicht da.

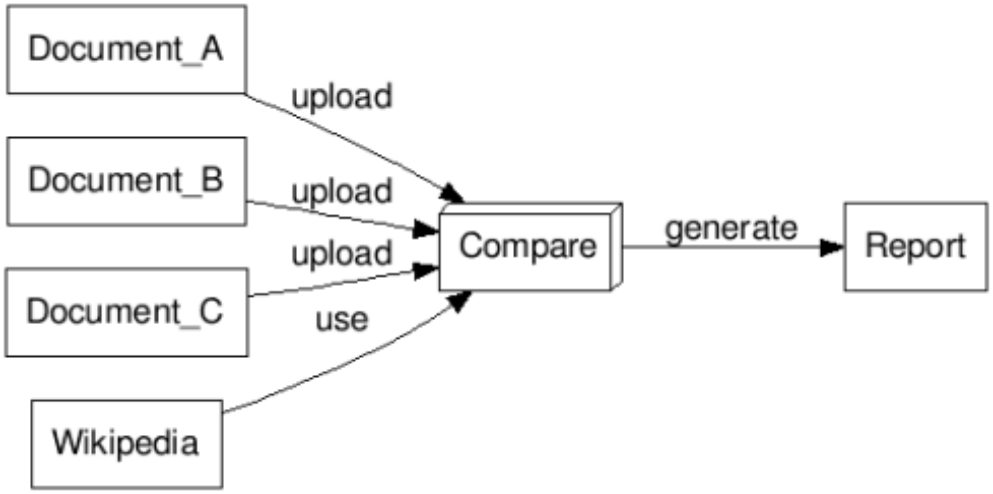
Gruppen

Sie werden in kleinen Gruppen von zwei bis vier Teilnehmern zusammen an einer Komponente arbeiten. Dabei müssen Sie sich mit den anderen Gruppen über Schnittstellen und Anforderungen abstimmen.

System

Stark vereinfachter (= naiver) Aufbau eines Plagiatserkennungssystems

(Wir werden die Architektur im Lauf



Ziele des Seminars

Ziel dieses Seminars ist, gemeinsam ein System zur Plagiatserkennung in wissenschaftlichen Arbeiten zu entwickeln. Dabei werden nicht alle Komponenten voll ausgestaltet.

Am Ende des Seminars werden Sie einen Einblick in folgende Themen haben:

- Verständnis von Plagiaten
- Algorithmen zur Ähnlichkeitssuche
- Arbeiten auf der Kommandozeile
- Text-Verarbeitung mit Python
- Versionskontrolle mit Git
- Praktische Erfahrung im Projektmanagement
- Verarbeiten von großen Datenmengen

Eventuell finden wir auch eine untere Schranke für die Dunkelziffer bei Plagiaten

Projektziele

Ein System, in das man vorne Dokumente reinsteckt, und wo hinten Plagiatsdokumentation rauskommt.

Konkrete Anforderungen:

- sehr modulares System
- soll mit mehreren hunderttausend Dokumenten umgehen können
- halbwegs effizient
- PDF-Dokumente, HTML und Plaintext einlesen
- irrelevante Inhalte (z.B. Seitenzahlen) filtern
- Zitate und Quellenangaben automatisch erkennen
- verschiedene Erkennungsalgorithmen parallel und einen kombinierten PlagScore bilden
- Interaktive Annotation von Plagiaten, die schlecht automatisiert erkannt werden können
- Visualisierung: Annotationen im Original-PDF einfügen, Graphen der Übereinstimmungen

zwischen Dokumenten

Anforderungen

Um das Seminar erfolgreich abzuschließen, wird von Ihnen folgendes erwartet:

- Sie halten ein kurzes Referat (5 Minuten) zu einem Teilaspekt des Seminars
- Sie sind regelmäßig im Seminar anwesend und diskutieren aktiv mit.
- Sie implementieren und dokumentieren in einer kleinen Gruppe eine Komponente für ein Plagiaterkennungssystem.
- Sie stellen ihre Arbeit bei der gemeinsamen Präsentation vor.

Zusätzlich würden wir Sie darum bitten, Ihre Beiträge unter einer Open-Source-Lizenz (CC-BY für Text und MIT license für Code) freizugeben.

Ablauf

- 19.04. Intro, Plagiate
- 26.04. Text-Verarbeitung auf der Kommandozeile
- 03.05. *Berufungsvortrag*
- 10.05. Überblick „Algorithmen zur Plagiatserkennung“, Text-Verarbeitung mit Python, Themenvergabe Referate
- 17.05. *entfällt (Christi Himmelfahrt)*
- 24.05. Vorstellung Algorithmen (Referate), Planung Projekte
  - 31.05. Systemarchitektur, Projektmanagement, Versionskontrolle
  - 07.06. *entfällt (Fronleichnam)*
  - 14.06. Präsentation Milestone 1
  - 21.06. Präsentation Milestone 2
  - 28.06. Präsentation Milestone 3
  - 05.07. Präsentation Milestone 4
  - 12.07. Release v0.1

Referate

Anforderungen:

- Handout auf Unterseite im Wiki, spätestens am Dienstag, 22.05.2012, 23:59 Uhr
- Vortrag, 10 Minuten, inkl. Folien/Tafelanschrift, etc. am 24.05.2012
- Demo-Implementierung d. Algorithmus in Python, inkl. Testdaten
- Besprechung mit Raphael vor dem Referat

Themen und Material: siehe Unterseite "Kurzvorträge"

Ansätze zur Plagiatserkennung

Aus der Sitzung vom 10.05.2012:

- gleiche Sätze in beiden Dokumenten, auch permutiert
- Stilanalyse
- Ähnlche Rechtschreibfehler / Rechtschreibung (Schweiz: 'ss' statt 'ß')
- Homonyme
- Synonyome
- Ähnliche Ideen
- Alle Dokumente in die gleiche Sprache übersetzen
- Grundformen der Wörter bilden (Stemming)
- unnötige Wörter entfernen (= Stoppwörter)

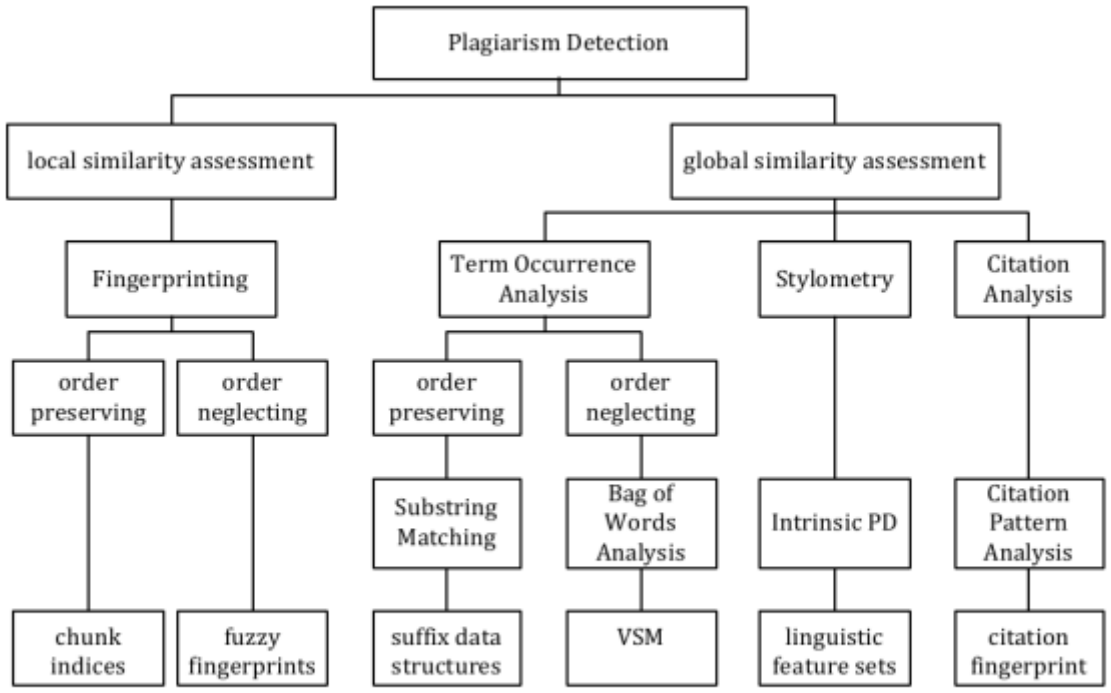


Figure 1: Classification of PD methods (inspired by: [47])

Quelle: Gipp, B. und Meuschke, N.: Citation Pattern Matching Algorithms for Citation-based Plagiarism Detection: Greedy Citation Tiling, Citation Chunking and Longest Common Citation Sequence. In Proc. ACM DocEng 2011

Infrastruktur

Arbeitsplätze

Donnerstag, 14-18 Uhr ist der CIP-Pool PHY 1.0.02 für uns reserviert. Auch in den restlichen Zeiten können Sie darin arbeiten, wenn gerade keine Lehrveranstaltung dort stattfindet. Sie können sich auch per SSH auf einem der CIP-Pool-Rechner einloggen. Wenn Sie größere Rechenjobs starten wollen, überprüfen Sie bitte vorher mit dem Befehl who, ob noch jemand anderes gerade an diesem Rechner arbeitet.

Speicherplatz

/home: In Ihrem Home-Verzeichnis auf dem Linux-Rechner haben Sie nur 50 MB frei. Das Home-Verzeichnis wird auf allen Rechnern über das Netzwerk eingebunden.

/localtmp:Auf jedem CIP-Pool_Rechner gibt es das Verzeichnis `/localtmp`. In diesem können Sie temporäre Dateien ablegen. Diese Verzeichnisse werden *nicht* mit zwischen den Rechnern synchronsiert und können jederzeit gelöscht werden.

/(noch nicht angelegt): In diesem globalen Projektverzeichnis befinden sich alle Daten. Es wird über das Netzwerk eingebunden. Details zur Datenstruktur folgen noch

Git Repository

Der komplette Programmcode - ohne Daten! - befindet sich in einem gemeinsamen Git Repository unter *(TODO)*. Verwenden Sie zur Authentisierung Ihren NDS-Account.

Es folgt noch:

- Anleitung zur Benutzung
- Richtlinien für Commits
- Dateistruktur

Wiki

Dieses Wiki dient dazu, das Seminar und die einzelnen Projekte zu dokumentieren. Legen Sie dazu bitte Unterseiten zu dieser Seite an, z.B. lehre:seminar_plagiate_ss_12:PDFParser.

Ressourcen

Plagiate:

- Raphaels Bookmarks zu Plagiaten
- GuttenPlag (siehe auch die dortigen Linksammlungen)
- VroniPlag

Unix shell scripting:

- Übersicht Unix-Tools für Textverarbeitung
- Advanced Bash Scripting Guide

Python:

- Python 2.7 Documentation
- Raphaels Bookmarks zu Python
- Learn Python the Hard Way (Tutorial)
- Python 2.7 Quick Reference

Hilfreiche Tools

Python:

- Flask - Web-Microframework
- RQ - Queues zur asynchronen Verarbeitung von Daten
- PDFMiner - Bibliothek zum Analysieren von PDF-Dateien
- Natural Language Toolkit - natural language processing and text analytics
- NetworkX - Bibliothek für Graphen und Visualisierungen

Korpora

- Wikipedia
- Wiley Major Reference Works
- Dissertationen in der DNB
- PubMed Central
- ACM Digital Library
- Arxiv
- CiteSeerX