External and Intrinsic Plagiarism Detection Using Vector Space Models

3^{rd} PAN Workshop/1^{st} PAN Competition

M. Zechner, M. Muhr, R. Kern, **M. Granitzer**

TU-Graz Knowledge Management Institute
Know-Center Graz

10. September 2009

Agenda

1. Extrinsic Plagiarism Detection
   - Overview
   - Approach
   - Experiments & Results
   - Open Issues
2. Intrinsic Plagiarism Detection
   - Overview
   - Approach
   - Experiments & Results
   - Open Issue

Motivation

- Goal: Identify document passages **partially derived** from other documents
  - Partially: document, section, paragraph or sentence
  - Derived: equal sequence, similar bag of words, similar phrases

Motivation

- Goal: Identify document passages **partially derived** from other documents
  - Partially: document, section, paragraph or sentence
  - Derived: equal sequence, similar bag of words, similar phrases
- PAN Corpus
  - Plagiarism on the passage level
  - Different obfuscation levels

Nearest Neighbor Search

- NN-search to identify *similar* passages

Nearest Neighbor Search

- NN-search to identify *similar* passages
- Curse of Dimensionality [Ind04]
  - Inverted Index exploits sparseness
  - Local Sensitive Hashing & Random Projections [GIM99]
  - Cluster Pruning [CPR$^+$07]

Nearest Neighbor Search

- NN-search to identify *similar* passages
- Curse of Dimensionality [Ind04]
  - Inverted Index exploits sparseness
  - Local Sensitive Hashing & Random Projections [GIM99]
  - Cluster Pruning [CPR+07]
- Feature Representation & Similarity Metric
  - Word n-grams on the document level [BR09]
  - Bag of Words (1-grams)

Nearest Neighbor Search

- NN-search to identify *similar* passages
- Curse of Dimensionality [Ind04]
    - Inverted Index exploits sparseness
    - Local Sensitive Hashing & Random Projections [GIM99]
    - Cluster Pruning [CPR$^+$07]
- Feature Representation & Similarity Metric
    - Word n-grams on the document level [BR09]
    - Bag of Words (1-grams)

Our Approach

Three main decisions:

- Bag-of-word representation on a sentence level
  - Identify at least one sentence in a plagiarized passage
  - Use bags for strongly obfuscated sentences
- Cluster pruning for speed-up
  - Balanced, similarity based partitioning via balanced on-line k-means [Zho05]
  - Find best partition first, then search within partition
- Post Processing to merge sentences to passages

Indexing Step

Given a set of reference documents D_r

Index

1. $\texttt{preprocessing}(D_r) \rightarrow S_r$
2. $\texttt{balancedOnlineKMeans}(S_r)$
   $\rightarrow C = \{C_1 \dots C_j\}, C_1 \cap C_2 = \emptyset$
3. $\texttt{store}(S_r, C_l)$

Why Clustering?

- Fast query time through balanced partitioning of the data set
- Random Clustering most probably achieves balanced partitioning on text data [CPR+07]
  - Our approached "ensured" balancing via threshold adaption
- Best runtime vs. accuracy trade-off achievable through hard clustering [CPR+07]
  - Some further heuristics for speeding up calculations
- Indexing requires two passes over all sentences/documents $O(|S_r| \cdot |C|)$

Retrieval Step

Given a suspicious documents D_s

Retrieve Plagiarized Sentences

1. `preprocess(`D_s`)` $\longrightarrow S_s$
2. `for every sentence` $s_i \in S_s$
    1. `lookupBestMatchingClusters(`s_i`)` $\longrightarrow \{C_m, C_n\}$
    2. `getKMostSimilarSentences(`C_m, C_n, k`)` $\longrightarrow S_c$
    3. `if` $\exists_{s_k \in s_c} cos(s_i, s_k) > \alpha$ `add` s_i `to the set of plagiarized sentences` S_p

Requires $O(|C| + k)$ evaluations per sentence

Postprocessing Step

Given a set of plagiarized sentences S_p

Merge Sentences to Passages

1. `for every plagiarized sentence` $s_i \in S_p$ `with a corresponding reference sentence` s_k^{ref}
    1. `if` $cos(s_{k+1}^{ref}, s_{i+1}) > \beta$ `then` $S_p = S_p \cup s_{i+1}$
    2. `if` $cos(s_{k-1}^{ref}, s_{i-1}) > \beta$ `then` $S_p = S_p \cup s_{i-1}$
2. `if two neighbor sentence are marked as plagiates, merge them`

Experimental Results

- Corpus statistic
  - $7 * 10^6$ reference sentences
  - $13 * 10^6$ suspicious sentences
- indexing took around 2h ($l = 50$, single core)
- lookup took around 2h ($k = 1$, single core()
- parameter study for k, l, α, β on a random sample of 500 suspicious documents

Experimental Results

l - k	*Prec.*	*Rec.*	*F1*	*Gran.*	*Rec. None*	*Rec. Low*
50 - 2	0.9616	0.4045	0.5695	1.9817	0.7044	0.4937
50 - 20	0.9523	0.4119	0.5750	1.9774	0.7053	0.4983
50 - 200	0.9411	0.4210	0.5818	1.9738	0.7053	0.5075
100 - 2	0.9597	0.4101	0.5746	1.9767	0.7044	0.4954
200 - 2	0.9419	0.4132	0.5745	1.9739	0.7050	0.4988
500 - 2000	0.8149	0.4782	0.6027	1.8497	0.7027	0.5534
Competition	0.6051	0.3714	0.4603	2.4424	-	-

- Trade-off speed vs. accuracy
- recall: l, k + sentence splitting
- precision: post processing

Open Issues

1. Accuracy of sentence splitting
2. Word n-grams as more discriminative features
3. Improvement over random projections
4. Larger blocks than sentences
5. Model selection during on-line clustering
6. Overlapping blocks resp. fixed block size

Motivation

- Goal: Identify sentence that differ significantly from the rest of the document
- Detect changes in style to do so ([MzES06],[Gri07])
- Hypothesis: plagiarized sentences differ significantly from average document style
  - Measure the similarity among styles
  - Style features to use
  - Combination of different style features

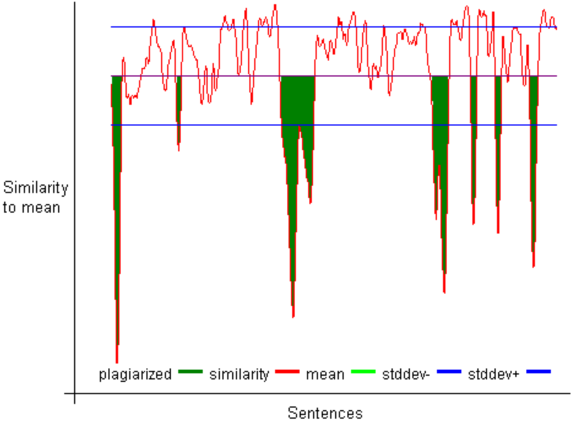
Approach

Given a suspicious documents D_s

- Calculate style vector for every sentence s_t
  - for all sentence in the window $s_t \pm l$
  - extract style features and their frequency
- Calculate document mean style vector for suspicious document $m = \frac{1}{N} \sum_{s_i \in D_s} s_i$
- for every sentence s_t
  - calculate cosine similarity $cos(s_t, m)$
  - Mark as plagiary if $cos(s_t, m) \leq \mu - \epsilon * \sigma$
- Merging as postprocessing step

Approach

Example



Features Used

- Word frequency class: $\lfloor \log(freq_{w*}/freq_w) \rfloor$
- Punctuation frequency
- Pronoun frequency
- POS-Tag Frequency
- Stopword Frequency

Experimental Results

Feature Space (k-l)	*Prec.*	*Rec.*	*F1*	*Gran*
Word Freq. Class (6-3)	0.2215	0.0934	0.1314	-
Punctuation (12-9)	0.1675	0.1908	0.1784	-
Part of Speech Tags (6-6)	0.1797	0.1791	0.1794	-
Pronouns (12-9)	0.1370	0.3587	0.1983	-
Closed Class Words (12-9)	0.1192	0.1467	0.1316	-
Combined Feature Space (12-6)	**0.1827**	**0.2637**	**0.2159**	-
Competition Corpus	0.1968	0.2724	0.2286	1.2942

Open Issues

- Optimizing weights for different style feature classes
- Supervised models

Thanks for your attention!
Questions?

Michael Granitzer
mgrani@know-center.at
http://www.know-center.at/

+43 316 873 9263

Alberto Barr and Paolo Rosso.
On automatic plagiarism detection based on n-grams comparison.
Advances in Information Retrieval, pages 696–700, 2009.

Flavio Chierichetti, Alessandro Panconesi, Prabhakar Raghavan, Mauro Sozio, Alessandro Tiberi, and Eli Upfal.
Finding near neighbors through cluster pruning.
In *PODS '07: Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on*

Principles of database systems, pages 103–112, New York, NY, USA, 2007. ACM.

- Aristides Gionis, Piotr Indyk, and Rajeev Motwani.
  Similarity search in high dimensions via hashing.
  In *VLDB '99: Proceedings of the 25th International Conference on Very Large Data Bases*, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

- Jack Grieve.
  Quantitative authorship attribution: An evaluation of techniques.

Lit Linguist Computing, 22(3):251–270, September 2007.

P Indyk.
Nearest neighbors in high-dimensional spaces.
Handbook of Discrete and Computational Geometry, 2004.

Sven Meyer zu Eissen and Benno Stein.
Intrinsic plagiarism detection.
In Mounia Lalmas, Andy MacFarlane, Stefan M. Rüger, Anastasios Tombros, Theodora Tsikrika, and Alexei Yavlinsky, editors, *ECIR*, volume 3936

of *Lecture Notes in Computer Science*, pages 565–569. Springer, 2006.

Shi Zhong.
Efficient online spherical k-means clustering.
In *Neural Networks, 2005. IJCNN '05. Proceedings. 2005 IEEE International Joint Conference on*, volume 5, pages 3180–3185 vol. 5, July-4 Aug. 2005.