Plagiarism Detection

CS533 Information Retrieval Project

İsmail UYANIK-Deniz KERİMOĞLU

Problem Description

- Plagiarism: The act of presenting another's work or ideas as your own.

- Avaliability of digital documents provides good chances for plagiarism.
- Types of plagiarism:
  - Copy-paste plagiarism
  - Paraphrasing
  - Translation
  - Idea Plagiarism

Motivation

- Plagiarism has turned into a serious problem for publishers, researches and educators.
- The main motivation is to protect the property rights of the owner.
- The task is to find all text passages in the suspicious document which have been plagiarized.

Methodology

- The algorithm consists of 3 steps
  - Selection: Reduces the search space by selecting a small number of suitable candidates for plagiarism.
    - Word length compression
    - N-gram distance
  - Matches: Performs detailed analysis on selected texts looking for matches longer than a fixed threshold (e.g. 15 characters)
    - T9 encoding
    - Longest common substring algorithm
  - Squares: Joins the set of plagiarised passages

Selection Phase

Aim: Reduce the search space to 10 most similar documents via a quick first look.

- First, the documents are compressed into a smaller size, where each word is converted into word length (Words larger than 9 characters are saturated to 9).
- The resulting alphabet consists of 9 symbols {1,2,….,9}.
- We obtain %82.5 compression ratio on the average.

Selection Phase

N-gram distance:

- We use 8-gram distance to detect similarities between suspicious and source documents.
- For a particular n-gram w, let f(w) denote the frequency of w and $D_n(x)$ is the set of all n-gram frequency in document x. n-gram distance between document x and y is calculated as follows:

$$d_n(x,y) := \frac{1}{|D_n(x)| + |D_n(y)|} \sum_{\omega \in D_n(x) \cup D_n(y)} \left(\frac{f_y(\omega) - f_x(\omega)}{f_y(\omega) + f_x(\omega)} \right)$$

- The most similar 10 source documents are passed to the second step.

Comparison Phase

- The idea is to look for common subsequences (matches) longer than a threshold, i.e., 15.
- First of all, we perform T9-like algorithm to translate different letters into the same character (e.g. {a,b,c}→2, {d,e,f}→3).
- The new alphabet for the coded text is made up of 9 symbols.
- The use of T9-like algorithm provides almost unique translation.

Comparison Phase

Longest common substring (LCS):

- The **longest common substring problem** is to find the longest string that is a substring of two or more strings. To identify the plagiarized passages we find the LCS.
- For the example strings "ABAB" and "BABA":

		A	B	A	B
	0	0	0	0	0
B	0	0	1	0	1
A	0	1	0	2	0
B	0	0	2	0	3
A	0	1	0	3	0

- The longest common substrings: "BAB","ABA".

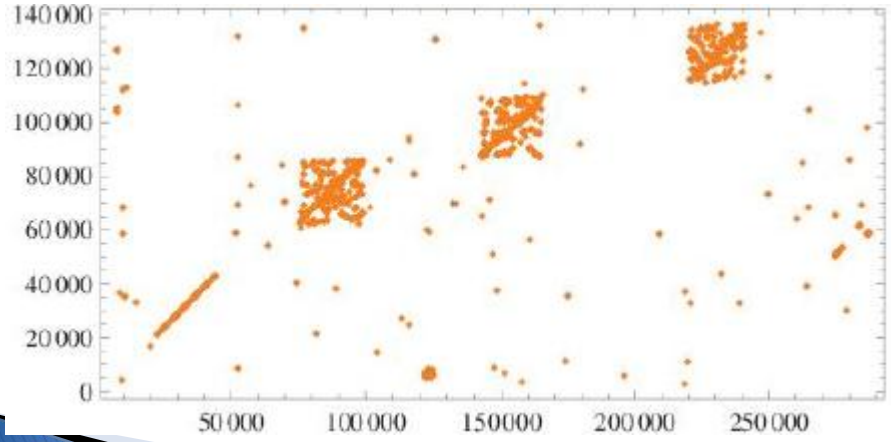
Comparison Phase

- We applied a different methodology for finding the plagiarized passages between 2 documents.
- LCS requires intensive computation power so we used fixed length chunks (windows) taken from document texts for LCS comparison. By setting the chunk size to 50 characters, the first chunk of suspicious document is compared with all chunks of the source document:

Merge Phase

Aim: Spot the obfuscated passages like copy-paste type plagiarism detection.

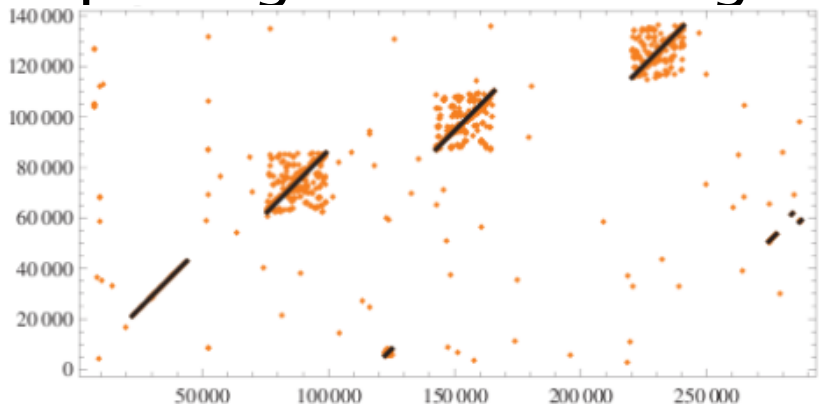
- When the plagiarism is of copy-paste type, it can be found in the comparison phase. But the obfuscation process, the act of plagiarism by changing the order of sentences, can be understood with a lot of small detections.



The copy-paste process can be seen as a line. Obfuscation process forms a square-like pattern.

Merge Phase

- A merging algorithm is applied for spotting plagiarized passages as in the figure.



- The merge is applied to matches from previous step if following conditions hold:
  - 1) Matches should be subsequent in suspicious document.
  - 2) The interval between matches in source and suspicious document is smaller than a threshold.
- By merging matches repeatedly, we obtain the resultant plagiarism passages that correspond to diagonals of square-like matches.

Implementation

- We implemented the word length compression and n-gram distance methods as described in the selection phase.
- We implemented T9-like encoding for the text passages and used windowing method before applying the longest common substring algorithm.
- We implemented the same merge algorithm defined in the paper. However, due to the use of windowing method we obtain a large number of small plagiarism actions and it is not easy to merge them to detect obfustucated plagiarism.
- We need parameter tuning for the new case..

Evaluation Measures

- We will use standard evaluation measures of IR, precision, recall, f-measure that are adapted for plagiarism detection problem.
- As a dataset, we will use the PAN-09 dataset for obtaining the evaluation results. Also, we want to make a comparison with results with Basile's paper.

PAN Plagiarism Corpus

- The PAN-PC-09 is a new large-scale dataset for the controlled evaluation of plagiarism detection algorithms.
- Corpus overview:
  - 41 223 text documents
  - 94 202 plagiarism cases
  - 70% is dedicated to external plagiarism detection,
  - 30% is dedicated to intrinsic plagiarism detection
  - Types of cases: monolingual with and without obfuscation, and cross-lingual
  - Authenticity of cases: real, emulated, and artificial

Conclusion

- We will test the three step algorithm which is run under PAN datasets and finally we expect to reveal most of the low level obfuscations and some of the low level obfuscations in the compared documents.

Thank you for listening…