CoReMo 2.1 Plagiarism Detector

Text Alignment Module

Diego A. Rodríguez Torrejón[1,2] José Manuel Martín Ramos[2]

[1]I.E.S. "José Caballero" dartsystems@gmail.com

[2]Universidad de Huelva (Spain) jmmartin@dti.uhu.es

*The **attendance** of Diego A Rodríguez Torrejón to CLEF2013 is **not sponsored or granted by any organization** (all helps refused). Its own company (Junta de Andalucia Educational Administration) will **cut off for 4 salary days** form Diego's pocket due to attending to the CLEF congress in laboral days.*

I hope don't "win" anymore. My family econonmy cannot support "vicious" research.

CoReMo will not be more a research product if cannot be substained by itself. In order to get fundings, CoReMo Web Services will be available soon as cheap (but non-free) detection services.

You can reach CoReMo Web Services at: ***http://www.coremodetector.com***

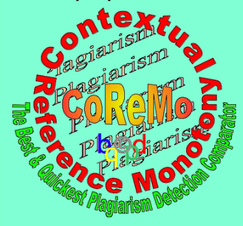
CoReMo Plagiarism Detector System

CoReMo is a crosslingual external focusing plagiarism detector system, created to participate in PAN competitions.

CoReMo has participated in PAN since 2010 to 2012 editions, renewed every time, looking for improved performance, high speed response and low hardware requirements.

The CoReMo effectiveness and its high speed are due to the combination of special N-grams type (xCTnG), a High Accuracy Information Retrieval Systems (HAIRS), a prune strategy to minimize comparisons (Reference Monotony), an integrated local translation system and, from now, a precise pairs document comparison module, everything joined to a hard C/C++ customized programming job.

CoReMo can be used in either command line or interface ways. A friendly web interface (***http://www.coremodetector.com***) is being finished to offer high quality / low cost analysis services very soon. Please, visit us to request free of charge evaluation while the startup phase.



Extended Contextual N-grams (xCTnG)

The documents are modeled by: case folding, stopwords and short length words removal, stemming and internal sort of n-gram components.

The unigrams obtained quadruple the natural n-grams due to the fact that a new special type of skip n-grams (Surrounding Context N-grams) and Odd/Even n-grams are also included in this CoReMo version.

Let's see how modelling xCTnG (CTnG + SC3G + OEnG)

"~~The~~ ***quick brown fox jumps*** ~~*over the*~~ ***lazy dog***"

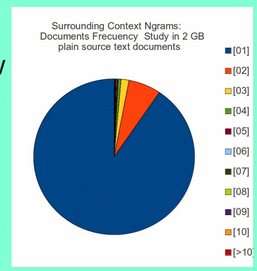
1. 1_2_3 QUICK BROWN FOX ⟶ BROWN_FOX_QUICK (CTnG)
2. 1_2_4 QUICK BROWN JUMPS ⟶ BROWN_JUMP_QUICK (SC3G)
3. 1_3_4 QUICK FOX JUMPS ⟶ FOX_JUMP_QUICK (SC3G)
4. 1_3_5 QUICK FOX LAZY ⟶ LAZ_FOX_QUICK (OEnG)
5. 2_3_4 BROWN FOX JUMPS ⟶ BROWN_FOX_JUMP (CTnG)
6. 2_3_5 BROWN FOX LAZY ⟶ BROWN_FOX_LAZ (SC3G)
7. 2_4_5 BROWN JUMPS LAZY ⟶ BROWN_JUMP_LAZ (SC3G)
8. 2_4_6 BROWN JUMPS DOG ⟶ BROWN_DOG_JUMP (OEnG)
9. 3_4_5 FOX JUMPS LAZY ⟶ FOX_JUMP_LAZ (CTnG)
10. 3_4_6 FOX JUMPS DOG ⟶ DOG_FOX_JUMP (SC3G)
11. 3_5_6 FOX LAZY DOG ⟶ DOG_LAZ_FOX (SC3G) ...

It implies more possibilities to tackle obfuscation cases with almost the same practical high precision in the process. The biggest number of terms obtained acts as a magnifier effect in the analysis. The memory requirements are obviously quadrupled and processing time almost doubled, but it improves dramatically the performance.

HAIRS

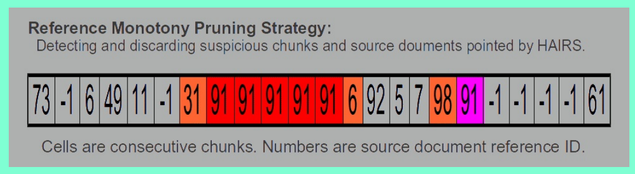
(High Accuracy Information Retrieval System)

Indexing the Extended Contextual Ngrams (about 90% has IDF = 1 in few GigaBytes/14000 docs. collections), a simplified SVM Information Retrieval System could get a highly discriminative detection of the most possible source for a short chunk of a suspicious text.



Reference Monotony Pruning Strategy

CoReMo uses this pruning way: *"discard matching if not happening monotonously"* in several modules and steps to avoid unneeded deep comparisons. An example is the combined action of HAIRS and the Reference Monotony Pruning strategy (RM): it doesn't detect plagiarism evidence by direct delimitation of the involved sections before making any comparisons. RM is also used for optimized web retrieval and discarding noisy matching in the detailed module.



Detailed Document Comparison (1) Trazable Ngram

Every Extendend Contextual N-gram obtained is used to generate a Trazable Ngram object, which registers the offset and length location, having two lists to register the n-grams matching ocurrences for inner and external cases.

TraceableNgram
ngram : string offset : long length : long <<list>> innerMatching : TraceableNgram <<list>> foreignMatching : TraceableNgram
compareTo(otherTraceableNgram : TraceableNgram) : int

Fastly Comparable Document

The document is modeled to get a fast comparison and a matching sections location method.

It uses two TrazableNgram vectors: the Natural vector (ordered by the natural position) and the Ordered vector (with former TrazableNgram references in alphabetical order, disambigued by position) got by QuickSort.

After getting the ordered version of the document, filling the inner matching list for every Trazable Ngram is fast and easy, having no more n-gram comparisons than the N-grams number existing in the largest document.

FastlyComparableDocument
<<vector>> NaturalVector : TraceableNgram <<vector>> OrderedVector : TraceableNgram wordLengthAverage : long
setMatchingTo(in otherDocument : FastlyComparableDocument) : void getDetectionInfo() : string

Detailed Document Comparison (2) Obtaining foreign matching

Having two Fastly Comparable Documents with inner matching lists annotations, its comparison is fast and easy, as there are the same n-gram comparisons that n-grams when the worst case happens.

When a matching is found by comparing progressively both ordered n-gram vectors in a MergeSort algorithm similar way, the inner matching list in a document will also be a foreing matching list for the other document, and the following comparison will advance one n-gram for every document. When no matching happens, the lower ordered n-gram will be the only advanced for the next comparison.

When a TrazableNgram is modified by the ordered list, it is consequently updated in the Natural order list.

Detection phase

Both inner and foreign lists annotations help to detect and locate the possible plagiarism by a distances and reliability combined methodology. The distances in the suspicious document are given as n-gram natural order distances. However, in the source document, they are arranged in character basics. For xCT3N mode, chunkLength and numNgrams are internally 4 times bigger.

$$wordLengthAverage = \frac{fileCharsLenth}{numNgrams}$$

$$maxNgramDist = 2 \cdot chunkLength$$

$$maxCharDist = chunkLength \cdot wordLengthAverage$$

$$minNgramLength = (monotony - 1.5) \cdot chunkLength$$

$$minCharLenght = minNgramLength \cdot wordLengthAverage$$

Granularity Reduction

After the direct detection (seeds), adjacent detections in both source and suspicious documents are joined by a "filtering phase" when the distance is lower than a threshold (4.000 chars), getting a significant improved performance when the lowest minimun length is required to consider a detection.

A comprehensive coloured comparison document could be created by combining both direct and filtered detections, detailling the evident direct detection and the joining effect.

Crosslingual Detection

When a non-English source document is detected, **CoReMo can use its own** word-by-word stem mapping **translation** locally for indexing or comparison purposes.

It gets a fastest response and high availaibility (and economy), as it is not foreign systems dependant.

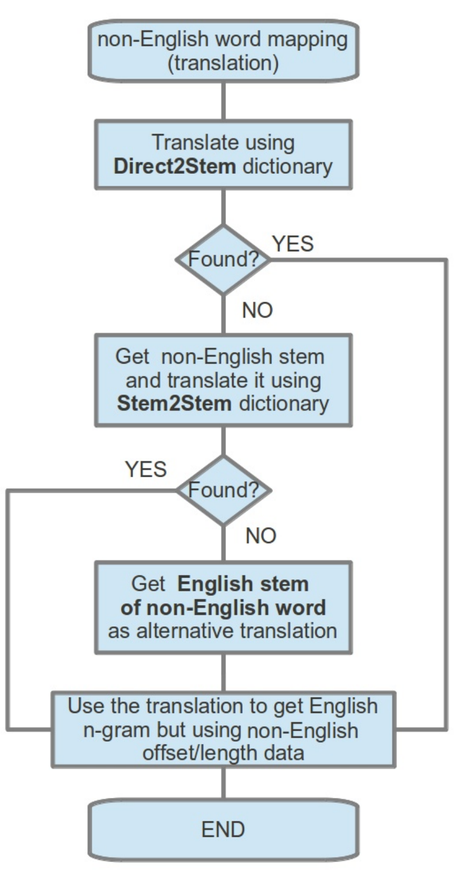
However, it can use improved quality translated versions of the documents from **external translation systems**.

This simple local translation mapping uses two special dictionaries (Direct2Stem and Stem2Stem) to get the quickest stem translation. The second one gets improved recall when the first one fails to find derivated words.

The system is currently using very poor dictionaries, and about 50% of words are not found at any: improved dictionaries should improve the results.

When getting the n-grams, the original non-English words offset and length are used, to get an exact location of the source section in the detection process.

Local Translation Method



A Fast Detection/Comparison Software

CoReMo has always been a highlighted fast software in former PAN editions. In PAN2013, it has been highlighted not only as **getting the best detections**, else as the fastest algorithm for the Text Alignment Task, being at least 4 times faster than any other competitor.

The keys for its remarkable speed are:

- Optimized C/C++ + OpenMP 64 bits programming
- GNU Linux 64bits OS and ext4 file system platform
- Internal sort of n-grams by Bubblesort algorithm
- n-grams into a document ordered by Quicksort
- Modified Mergesort algorithm to compare both docs
- Local translations by dictionary mapping
- Taking the advantage of suspicious document modelling when repeated in consecutive comparisons

However, this version takes the oportunity of **multicore** processors technology advantage, and a web interface.

The **runtime** for competition was tested by only a Single Core 2.27 GHz Virtual Machine, needing 75 seconds to analyze **5185 documents pairs**.

Using a 8 cores AMD FX8120 @ 4.0 GHz, we got the same analysis in **less than 5 seconds !!**.

CoReMo 2.1 performance

The table below shows the achieved performance in the PAN2013 Detailed Comparison Training corpus:

	PAN-PC-2013 Training Corpus			
	Plagdet	*Recall*	*Precision*	*Granularity*
No obfuscation	0.92733	0.97326	0.88554	1.00000
Random obfus.	0.75527	0.63388	0.93417	1.00000
Translated obfus.	0.84683	0.79951	0.90001	1.00000
Summary obfus.	0.35513	0.22973	0.87716	1.03529
Global				
Global bug fixed[3]	**0.82722**	0.76758	0.89929	1.00169

In the Text Alignment Task competition, CoReMo got the **1st Plagdet ranked**, with the best Global detections, and shining as having an excellent Precision.

Runtime for CoReMo was the best, far from any other.

	PAN-PC-2013 Competition Corpus				
	Plagdet	*Recall*	*Precision*	*Granularity*	*runtime (ms)*
No obfuscation	0.92586	0.95256	0.90060	1.00000	
Random obfus.	0.74711	0.63370	0.90996	1.00000	
Translated obfus.	0.85113	0.81124	0.89514	1.00000	
Summary obfus.	0.34131	0.21593	0.90750	1.07742	
Global	**0.82220**	0.76190	0.89484	1.00141	**72508**
Global bug fixed[3]	**0.82827**	0.77177	0.89564	1.00140	79965

Both performances were achieved by the parameters Monotony -> 2 chunks, and self-tuned chunk Length -> 4 to 8 natural n-grams (about 14 words including stopwords) for monolingual and 45 natural n-grams for non-English docs.