A plagiarism detection procedure in three steps: selection, matches and “squares”

Chiara Basile - basile@dm.unibo.it

Mathematics Department
University of Bologna, Italy

PAN‘09 Workshop, San Sebastián - Donostia, 10/09/2009

Joint work with Dario Benedetto, Emanuele Caglioti, Giampaolo Cristadoro, Mirko Degli Esposti

Once upon a time...

03/05/09

A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

Once upon a time...

03/05/09

A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09

Once upon a time...

03/05/09
A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09 - just one month...

Once upon a time...

03/05/09
A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09 - just one month...
...and a few documents: “just” 14,428!

Once upon a time...

03/05/09

A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09 - just one month...
...and a few documents: “just” 14,428!

Therefore, two imperatives:

Once upon a time...

03/05/09
A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09 - just one month...
...and a few documents: “just” 14,428!

Therefore, two imperatives:

1. be (not only computationally) fast

Once upon a time...

03/05/09

A group of mathematicians from the Universities of Bologna and Rome *La Sapienza* gets to know of the Plagiarism Competition and decides to try some preliminary experiments on the external plagiarism corpus using methods developed for different tasks, like authorship recognition and text categorization.

The competition deadline: 07/06/09 - just one month...
...and a few documents: “just” 14,428!

Therefore, two imperatives:

1. be (not only computationally) fast
2. use heuristics

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

The Gramsci Project

C. Basile, D. Benedetto, E. Caglioti, M. Degli Esposti
An example of mathematical authorship attribution
Journal of Mathematical Physics **49**, 125211 (2008).

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

and usually defining some similarity metric(s) to estimate the “distance” between couples of sequences.

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

and usually defining some similarity metric(s) to estimate the “distance” between couples of sequences.

Given two texts x, y their n-gram distance is:

$$d_n(x,y) := \frac{1}{|D_n(x)| + |D_n(y)|} \sum_{\omega \in D_n(x) \cup D_n(y)} \left(\frac{f_x(\omega) - f_y(\omega)}{f_x(\omega) + f_y(\omega)} \right)^2$$

where:

- $f_x(\omega)$ = frequency of the (character) $n-$gram ω in x;
- $D_n(x)$ = set of all the $n-$grams with non-zero frequency in x.

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

and usually defining some similarity metric(s) to estimate the “distance” between couples of sequences.

Given two texts x, y their n-gram distance is:

$$d_n(x,y) := \frac{1}{|D_n(x)| + |D_n(y)|} \sum_{\omega \in D_n(x) \cup D_n(y)} \left(\frac{f_x(\omega) - f_y(\omega)}{f_x(\omega) + f_y(\omega)} \right)^2$$

where:

- $f_x(\omega)$ = frequency of the (character) $n-$gram ω in x;
- $D_n(x)$ = set of all the $n-$grams with non-zero frequency in x.

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

and usually defining some similarity metric(s) to estimate the "distance" between couples of sequences.

Given two texts x, y their n-gram distance is:

$$d_n(x,y) := \frac{1}{|D_n(x)| + |D_n(y)|} \sum_{\omega \in D_n(x) \cup D_n(y)} \left(\frac{f_x(\omega) - f_y(\omega)}{f_x(\omega) + f_y(\omega)} \right)^2$$

where:

- $f_x(\omega)$ = frequency of the (character) $n-$gram ω in x;
- $D_n(x)$ = set of all the $n-$grams with non-zero frequency in x.

Where do we come from?

Various problems of classification and clustering of symbolic sequences (authorship attribution, classification of biological or genetic sequences, ...)

faced using ideas coming from Information Theory, Dynamical Systems, Statistical Mechanics...

and usually defining some similarity metric(s) to estimate the "distance" between couples of sequences.

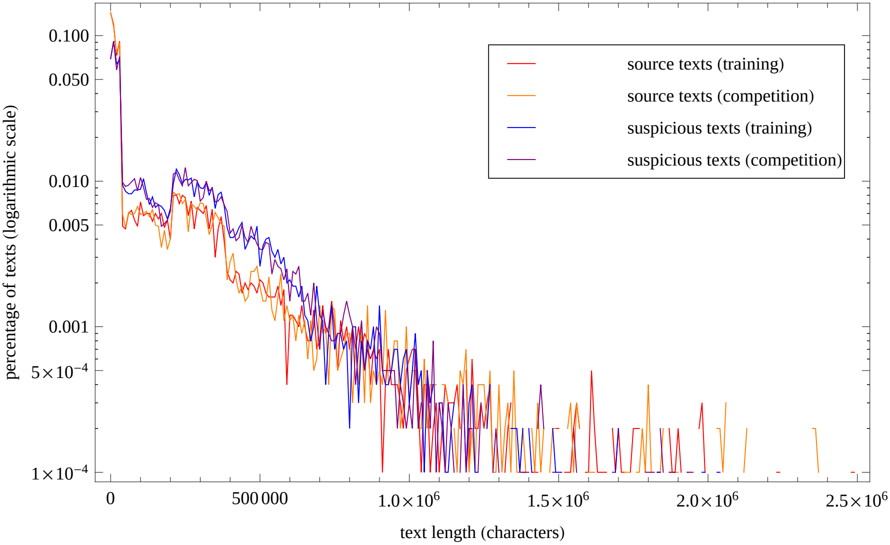
Given two texts x, y their n-gram distance is:

$$d_n(x,y) := \frac{1}{|D_n(x)| + |D_n(y)|} \sum_{\omega \in D_n(x) \cup D_n(y)} \left(\frac{f_x(\omega) - f_y(\omega)}{f_x(\omega) + f_y(\omega)} \right)^2$$

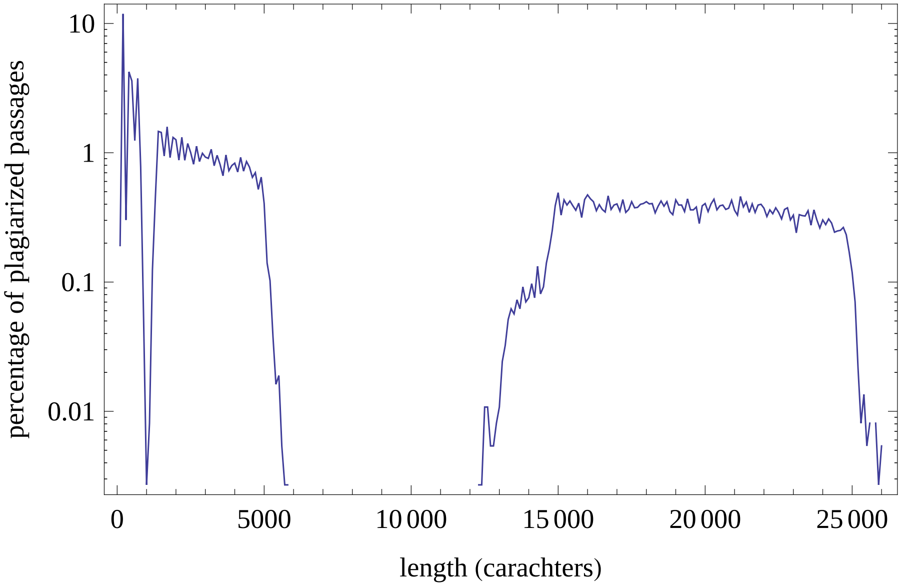
where:

- $f_x(\omega)$ = frequency of the (character) $n-$gram ω in x;
- $D_n(x)$ = set of all the $n-$grams with non-zero frequency in x.

Corpus statistics



Corpus statistics



1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the “normal” alphabet + it takes too long

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the “normal” alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the "normal" alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

We converted all texts into word lengths (up to 9):

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the “normal” alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

We converted all texts into word lengths (up to 9):

```
To be or not to be:  that is the question
```

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the "normal" alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

We converted all texts into word lengths (up to 9):

`To be or not to be: that is the question` $\rightarrow$ **`2223224238`**

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the “normal” alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

We converted all texts into word lengths (up to 9):

`To be or not to be: that is the question` $\rightarrow$ `2223224238`

The value $n = 8$ was chosen as a compromise between

- acceptable computational time (2.3 days for the whole corpus)

1 - Selection

First of all: reduce the search space by selecting a small number of suitable candidates for plagiarism for each plagiarized text.

Can we use the $n-$gram distance for this task?

Maybe, but there is not enough statistics using the “normal” alphabet + it takes too long $\Rightarrow$ reduce the alphabet!

We converted all texts into word lengths (up to 9):

`To be or not to be: that is the question` $\rightarrow$ **`2223224238`**

The value $n = 8$ was chosen as a compromise between

- acceptable computational time (2.3 days for the whole corpus)
- a good recall (81% of the plagiarized characters come from the first 10 neighbours $\rightarrow$ very good! 13% of translated plagiarism...)

2 - Matches

Now we can perform a detailed analysis on the 7214 x 10 couples of texts, looking for common subsequences (matches) longer then a fixed threshold (e.g. 15 characters).

2 - Matches

Now we can perform a detailed analysis on the 7214 x 10 couples of texts, looking for common subsequences (matches) longer then a fixed threshold (e.g. 15 characters).

A new conversion: T9 encoding.

2 - Matches

Now we can perform a detailed analysis on the 7214 x 10 couples of texts, looking for common subsequences (matches) longer then a fixed threshold (e.g. 15 characters).

A new conversion: T9 encoding.

Why T9?

- “almost unique” translation for long enough sequences (10-15 characters);

2 - Matches

Now we can perform a detailed analysis on the 7214 x 10 couples of texts, looking for common subsequences (matches) longer then a fixed threshold (e.g. 15 characters).

A new conversion: T9 encoding.

Why T9?

- "almost unique" translation for long enough sequences (10-15 characters);
- it reduces the alphabet to 10 symbols $\Rightarrow$ speeds up the indexing phase of the matching algorithm.

more...

2 - Matches

Now we can perform a detailed analysis on the 7214 x 10 couples of texts, looking for common subsequences (matches) longer then a fixed threshold (e.g. 15 characters).

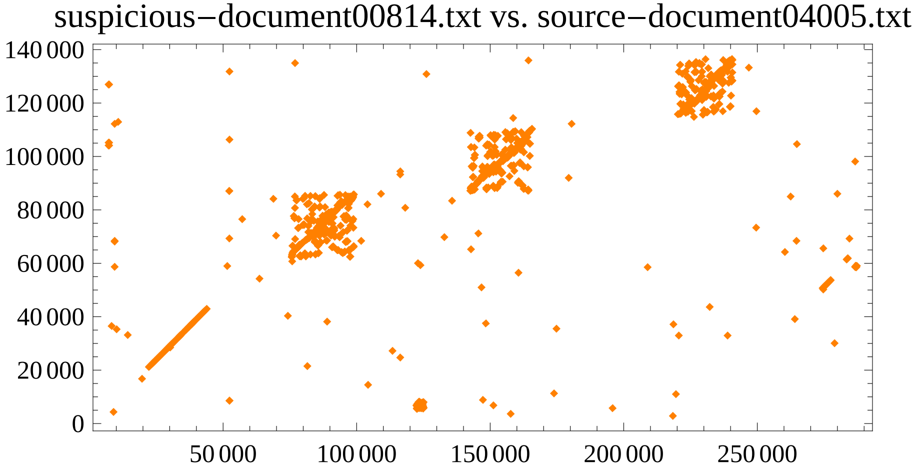
A new conversion: T9 encoding.

Why T9?

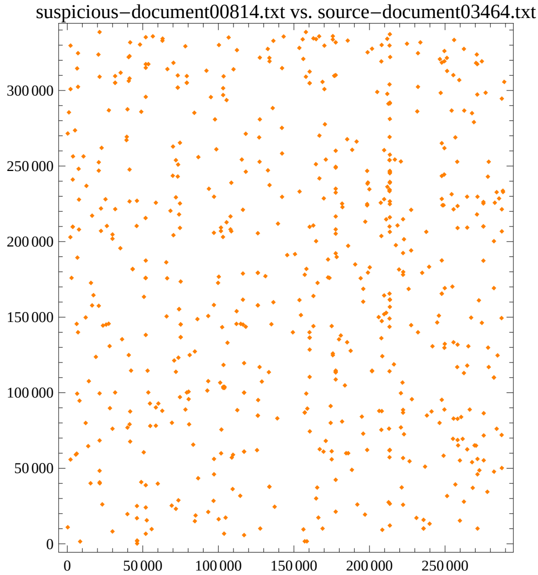
- "almost unique" translation for long enough sequences (10-15 characters);
- it reduces the alphabet to 10 symbols $\Rightarrow$ speeds up the indexing phase of the matching algorithm. ▸ more...

Computation times for the whole corpus: 40 hours.

2 - Matches (continued)

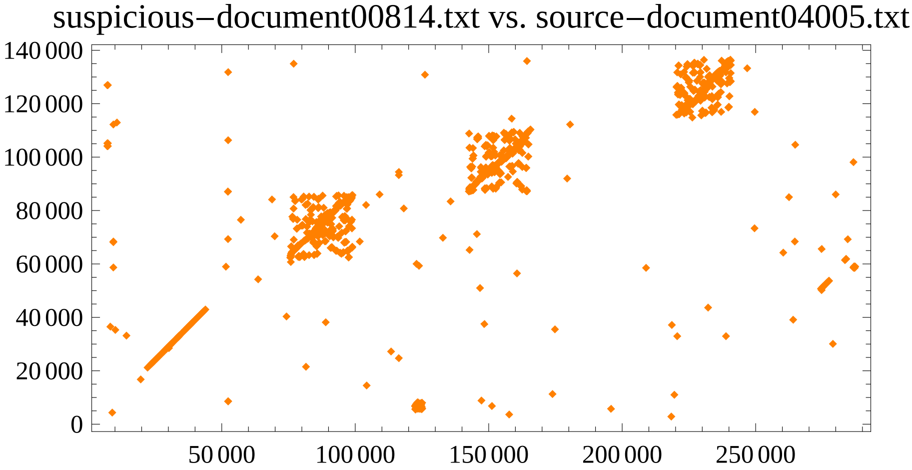


2 - Matches (continued)



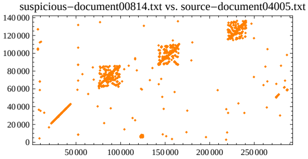
3-“Squares”

How to identify the “squares” which are so evident in this picture?



3-“Squares”

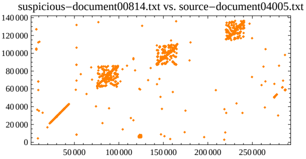
How to identify the “squares” which are so evident in this picture?



We need scalability!

3-“Squares”

How to identify the “squares” which are so evident in this picture?



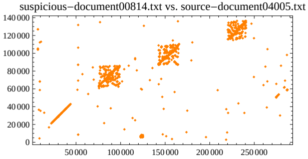
We need scalability!

Join two matches if the following conditions hold simultaneously:

1. matches are subsequent in the suspicious file

3-“Squares”

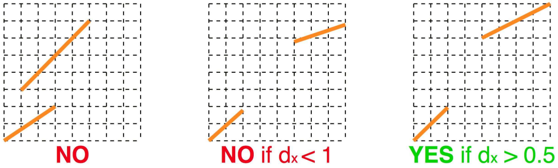
How to identify the “squares” which are so evident in this picture?



We need scalability!

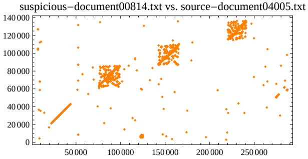
Join two matches if the following conditions hold simultaneously:

1. matches are subsequent in the suspicious file
2. matches are not superimposed in the suspicious file and their distance in the suspicious file is not larger than the length of the longest of the two sequences, scaled by δ_x



3-"Squares"

How to identify the "squares" which are so evident in this picture?



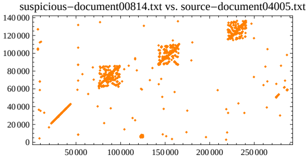
We need scalability!

Join two matches if the following conditions hold simultaneously:

1. matches are subsequent in the suspicious file
2. matches are not superimposed in the suspicious file and their distance in the suspicious file is not larger than the length of the longest of the two sequences, scaled by δ_x
3. the same as 2 (with possibly a different δ_y) in the source file

3-“Squares”

How to identify the “squares” which are so evident in this picture?



We need scalability!

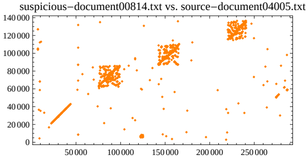
Join two matches if the following conditions hold simultaneously:

1. matches are subsequent in the suspicious file
2. matches are not superimposed in the suspicious file and their distance in the suspicious file is not larger than the length of the longest of the two sequences, scaled by δ_x
3. the same as 2 (with possibly a different δ_y) in the source file

Then: repeatedly merge superimposed segments

3-"Squares"

How to identify the "squares" which are so evident in this picture?



We need scalability!

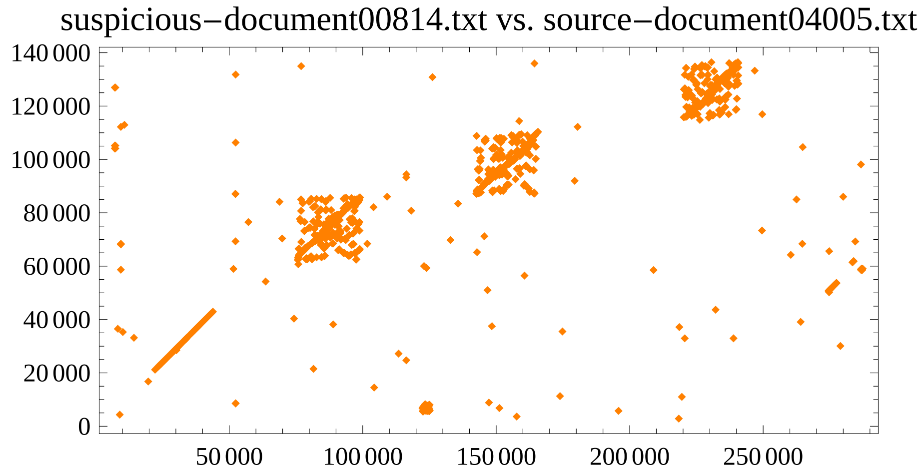
Join two matches if the following conditions hold simultaneously:

1. matches are subsequent in the suspicious file
2. matches are not superimposed in the suspicious file and their distance in the suspicious file is not larger than the length of the longest of the two sequences, scaled by δ_x
3. the same as 2 (with possibly a different δ_y) in the source file

Then: repeatedly merge superimposed segments
+ run the algorithm above again with smaller parameters δ'_x and δ'_y.

3-“Squares”

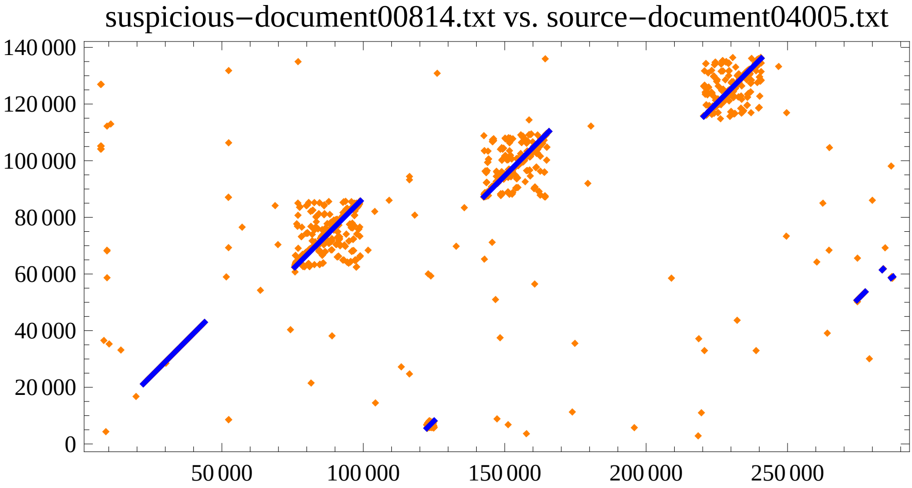
How to identify the “squares” which are so evident in this picture?



suspicious−document00814.txt vs. source−document04005.txt

3-"Squares"

How to identify the "squares" which are so evident in this picture?



Summary of the procedure

1 - Selection

2 - Matches

3 - “Squares”

Summary of the procedure

1 - Selection

`The Constance letters of Charles Chapin, edited by Eleanor Early and Constance...` $\longrightarrow$ `397276627539...`

Summary of the procedure

1 - Selection

```
The Constance letters of
Charles Chapin, edited by
Eleanor Early and
Constance...
```

$\longrightarrow$ `397276627539...`

$\Downarrow$ by the 8-gram distance $\Downarrow$

suspicious-document00814 $\left\{ \right.$

1) source-document04005
2) source-document04080
3) source-document02123
4) source-document02648
5) source-document03464
6) source-document02737
7) source-document03876
8) source-document05012
9) source-document04456
10) source-document04223

Summary of the procedure

1 - Selection

2 - Matches

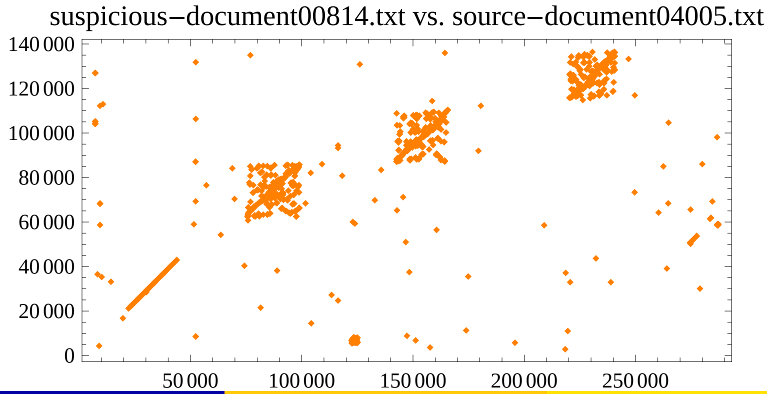
```
The Constance letters of          843026678262305388377O
Charles Chapin, edited by    ⟶    630242753702427461O
Eleanor Early and                 33483302903532667O32759O
Constance...                      2630266782623...
```

Summary of the procedure

1 - Selection

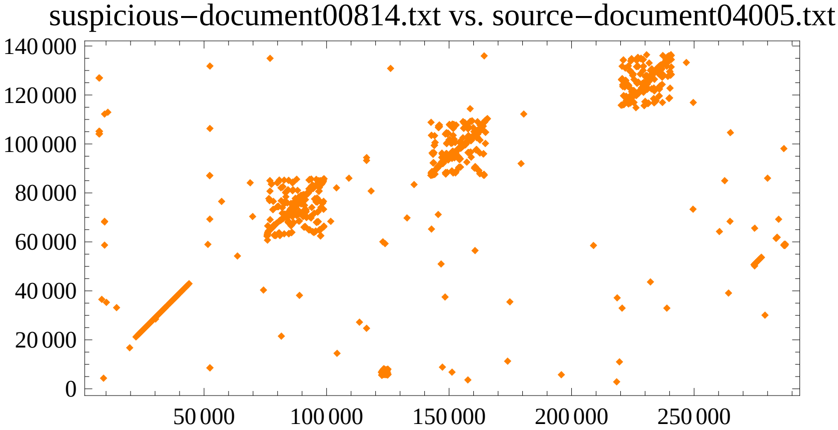
2 - Matches

```
The Constance letters of        84302667826230538837700
Charles Chapin, edited by   ⟶  630242753702427461O
Eleanor Early and               33483302903532667032759O
Constance...                    2630266782623...
```



Summary of the procedure

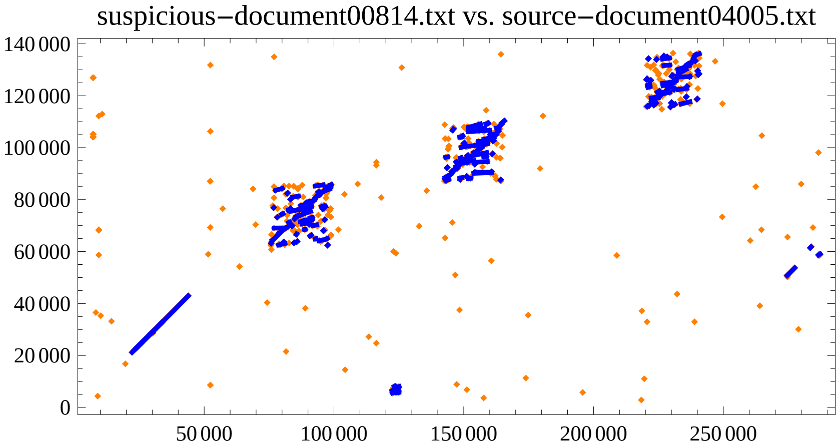
1 - Selection
2 - Matches
3 - “Squares”



1496 matches

Summary of the procedure

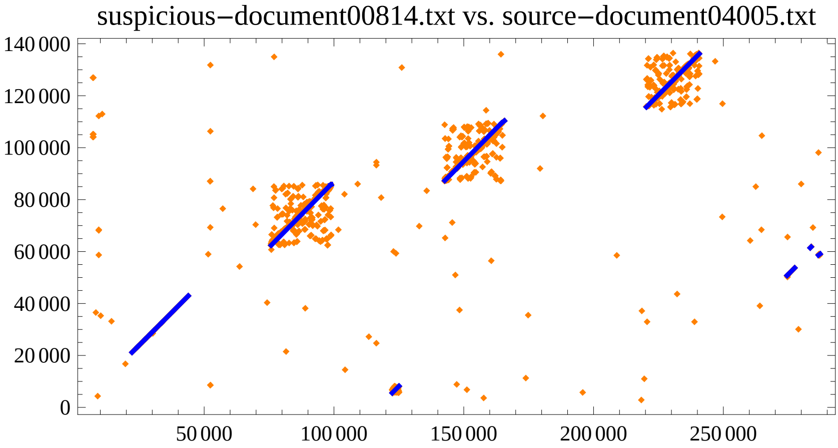
1 - Selection
2 - Matches
3 - “Squares”



1496 matches → 244 pieces

Summary of the procedure

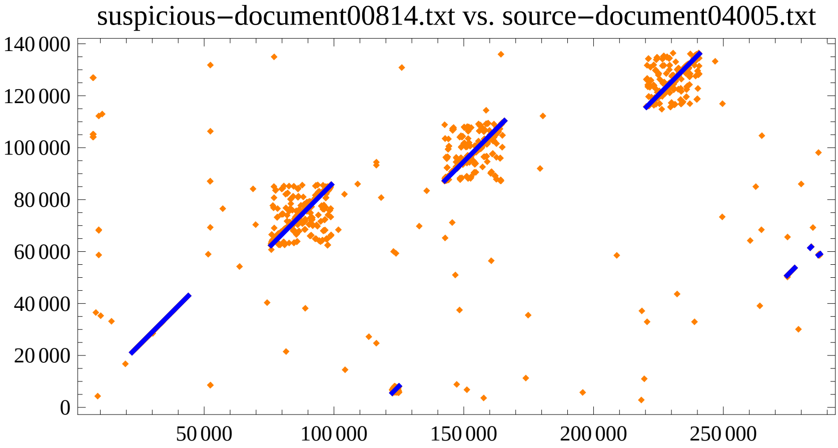
1 - Selection
2 - Matches
3 - “Squares”



1496 matches → 244 pieces→ 16 passages

Summary of the procedure

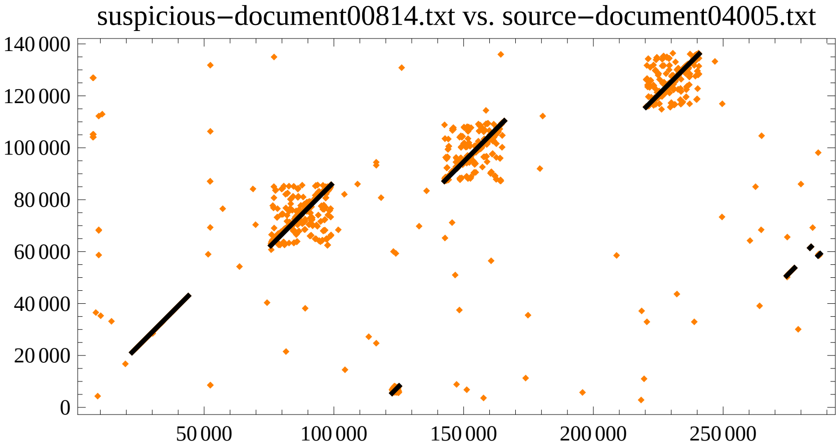
1 - Selection
2 - Matches
3 - “Squares”



1496 matches → 244 pieces→ 16 passages → 8 suspected plagiarisms

Summary of the procedure

1 - Selection
2 - Matches
3 - “Squares”



Comparison with the associated xml file... ok!

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

- less heuristics in the tuning of $\delta_x, \delta_y, \delta'_x, \delta'_y$... density of matches?

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

- less heuristics in the tuning of $\delta_x, \delta_y, \delta'_x, \delta'_y$... density of matches? Maybe they can be used to control precision, recall and granularity according to the task...

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

- less heuristics in the tuning of $\delta_x, \delta_y, \delta'_x, \delta'_y$... density of matches? Maybe they can be used to control precision, recall and granularity according to the task...
- there are certainly better ideas for the selection phase...

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

- less heuristics in the tuning of $\delta_x, \delta_y, \delta'_x, \delta'_y$... density of matches? Maybe they can be used to control precision, recall and granularity according to the task...
- there are certainly better ideas for the selection phase...
- try other/standard clustering algorithms

Results and conclusions

Results on the competition corpus, with $\delta_x = \delta_y = 3, \delta'_x = \delta'_y = 0.5$:

- Precision: 0.6727
- Recall: 0.6272
- F-measure: 0.6491
- Granularity: 1.0745
- Overall score: 0.6041

i.e. the third overall score after 0.6093 and 0.6957 of the first two.

Many possible improvements:

- less heuristics in the tuning of $\delta_x, \delta_y, \delta'_x, \delta'_y$... density of matches? Maybe they can be used to control precision, recall and granularity according to the task...
- there are certainly better ideas for the selection phase...
- try other/standard clustering algorithms

And... what about the internal plagiarism problem?

To conclude

Thank you!

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

index has length N and its i^{th} element is the index of the previous occurrence in s of the 7-gram $s_i, \ldots, s_{i+6}$

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

- **index** has length N and its i^{th} element is the index of the previous occurrence in s of the 7-gram $s_i, \ldots, s_{i+6}$
- **last** has length 10^7 and its j^{th} element is the index of the last occurrence of the 7-gram j (padded with zeroes on the left, if needed) in s

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

- **index** has length N and its i^{th} element is the index of the previous occurrence in s of the 7-gram $s_i, \ldots, s_{i+6}$
- **last** has length 10^7 and its j^{th} element is the index of the last occurrence of the 7-gram j (padded with zeroes on the left, if needed) in s

N.B. The minimum length for detected matches is 7

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

- **index** has length N and its i^{th} element is the index of the previous occurrence in s of the 7-gram $s_i, \dots, s_{i+6}$
- **last** has length 10^7 and its j^{th} element is the index of the last occurrence of the 7-gram j (padded with zeroes on the left, if needed) in s

N.B. The minimum length for detected matches is 7

Phase 2: every suspicious document t (length M) is ran through once and for each $k = 0, \dots, M-1$ the indexes $p = last(t_k, \dots, t_{k+6})$ and $index(p)$ are used to retrieve the position of the possible matches in s without running through it again.

Our matching algorithm

Phase 1: every source document s of length N is indexed (once and for all) by two vectors:

- **index** has length N and its i^{th} element is the index of the previous occurrence in s of the 7-gram $s_i, \ldots, s_{i+6}$
- **last** has length 10^7 and its j^{th} element is the index of the last occurrence of the 7-gram j (padded with zeroes on the left, if needed) in s

N.B. The minimum length for detected matches is 7

Phase 2: every suspicious document t (length M) is ran through once and for each $k = 0, \ldots, M-1$ the indexes $p = \mathit{last}(t_k, \ldots, t_{k+6})$ and $\mathit{index}(p)$ are used to retrieve the position of the possible matches in s without running through it again.

Total cost: $M + N$ for each couple suspicious-source.

back