San Jose State University

SJSU ScholarWorks

Master's Projects

Master's Theses and Graduate Research

Fall 2015

Author recognition using Locality Sensitive Hashing & Alergia (Stochastic Finite Automata)

Prashanth Sandela

San Jose State University

Author recognition using Locality Sensitive Hashing & Alergia (Stochastic Finite Automata)

A Thesis

Presented to The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

By

Prashanth Sandela

December 2015

San Jose State University

The Designated Thesis Committee Approves the Thesis Titled

Author recognition using Locality Sensitive Hashing & Alergia (Stochastic Finite Automata)

By

Prashanth Sandela

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE
SAN JOSÉ STATE UNIVERSITY

December 2015

Dr. T. Y. Lin, Department of Computer Science Date

Dr. Thomas Austin, Department of Computer Science Date

Mr. Karthik Rajagopalan, Lead Data Engineer at Target Date

ABSTRACT

In today's world data grows very fast. It is difficult to answer questions like 1) Is the content completely written by this author, 2) Did he get few sentences or pages from another author, 3) Is there any way to identify actual author. There are many plagiarism software's available in the market which identify duplicate content. It doesn't understand writing pattern involved. There is always a necessity to make an effort to find the original author. Locality sensitive hashing is one such standard for applying hashing to recognize authors writing pattern.

ACKOWLEDGEMENTS

I want to take to thank everyone who has contributed towards completion of my project. The journey towards the end of the project was fruitful, exciting and great learning procedure. Experience gained from the project will help me solve many problems in the future.

I would like to thank Dr. T.Y.Lin, my project advisor, for giving me this wonderful project and helping me to successfully complete this project and get a great insight about real time challenges and problems. Thanks for your attention and support. I would like to thank my committee members, Dr. Thomas Austin and Karthik Rajagopalan for support and patience.

I would like to thank Computer Science department for providing me opportunity and with the necessary material. I would like to thank everyone who has helped and guided me to successfully complete this project.

Contents

List of Tables

List of Figures

1. Introduction

There are many authors in today’s world. This count keeps growing day by day. It is estimated that there are more than 20 Billion authors. Anyone who writes some content which may be a book or blog post or a blog comment is considered as an author.

With the growing authors and post or books, it is being increasingly difficult to track authors identity. Even now there are millions of documents without a recognized author. It is important to know the true identity of the author so that you might want to read more of his documents. With the increasing access to internet, it is very easy to duplicate particular content and hide the real author. There are many plagiarism software’s available in the market but they can say if the content is original, they don’t have the capability to identify author. There is really a necessity to have a model which tells the author. Our model should be able to identify author of the document.

The main basis to identify the pattern of the author is very different from other types of mining. For many types of data mining, we eliminate most of the stop words and special characters to make a sense out of data. For recognizing authors writing pattern we have to consider stop words and special characters, helps identify sentences. This can be achieved using many hashing algorithms of Locality Sensitive Hashing [1][2] and using Stochastic Finite Automata Technique (Alergia Algorithm).

2. Locality Sensitive Hashing

Locality Sensitive Hashing [2] reduces dimensionality of high-dimensional data. It has a hashing method which has high probability of mapping similar items to same buckets i.e., the total number of input items are much higher than the no. of buckets. This hashing is different from conventional hash functions and cryptographic hash functions coz it aims to maximize the possibility of like-items-collision. This Hashing is commonly used in data clustering and nearest neighbor search.

2.1. Definition

Locality Sensitive Hashing family [5][6][7] **F** is defined for metric space **M,** an approximate factor **C** and a threshold **R**.

$$\mathbf{M = (M, d), R > 0, C>1}$$

The **F** is a family of functions **h : M => S** which maps to the elements of **M** to a bucket **s ϵ S**.

Locality Sensitive Hashing family satisfies below conditions for **p, q ϵ M**, using **h ϵ F**

- If **d(p,q) ≤ R**, then **h(q) equals h(p)** i.e.., **p** and **q** collides with probability > $\mathbf{P_1}$
- If **d(p,q) ≥ CR**, then **h(q) equals h(p)** i.e.., **p** and **q** collides with probability < $\mathbf{P_2}$

A family when $\mathbf{P_1}$ > $\mathbf{P_2}$. Then this **F** is called $\mathbf{(R, cR, P_1, P_2)}$

LSH is defined with universe U with a similarity function **Ø : U X U -> [0, 1].**

This locality sensitive hashing scheme is a family of **H** which are paired with probability distribution **D,** such that a h ϵ H chosen according to **D**.

$$\mathbf{Pr_{h \epsilon H}[h(a) = h(b)] = Ø\ (a,b)} \text{ for any } \mathbf{a, b \ \epsilon \ U}$$

2.2. Applications

- Image similarity identification
- Hierarchical clustering
- Audio similarity identification
- Near-duplicate detection
- Genome-wide association study
- Audio fingerprint
- Nearest neighbor search
- Digital video fingerprinting

2.3. Hashing algorithms following Locality Sensitive Hashing

- Min-Hashing
- Sim-Hashing
- Hamming Similarity
- Angle Similarity

3. MinHash

MinHash [16] hashing scheme is a technique to estimating similarity between two sets. MinHash is also known as Min-Wise Independent permutations of Locality Sensitive Hashing. Andrei Broder invented MinHash scheme. In the early stages of search engine, this scheme was used in AltaVista search engine to detecting duplicate web pages. MinHash has its application in largest-scale clustering documents or problems.

3.1. Jaccard Similarity and minimum hash values

The Jaccard similarity coefficient[16] is also called Jaccard index. This is used to indicate similarity of two sets.

Let us consider 2 sets A and B.

A ∩ B => intersection of 2 sets

A U B => union of 2 sets.

Jaccard similarity coefficient can be defined as ratio of intersection and union of sets.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

The value of Jaccard similarity coefficient is strictly between 0 and 1.

J (A,B)=0 if the sets are disjoint, (No common elements)

=1 if the sets are equal

<=1 if the sets have relatively more members in common.

The goal of MinHash is to quickly estimate J(A,B) without performing any operations like union and intersection.

Let us assume a hash function **h**. Let this map X and Y to distinct integers. Set Z defines $h_{\min}(Z)$ to be least member of Z with regards to h i.e.., the member n of Z with least value of h(x). If $h_{\min}$ is applied to X and Y, it can be observed that they get same values when elements of (X U Y) with a minimum hash value which resides in (X ∩ Y).

> Pr[$h_{\min}(X) = h_{\min}(Y)$] = $J(X,Y)$, This equation implies that this follows Locality Sensitive Hashing.

This implies that the probability of $h_{\min}(X) = h_{\min}(Y)$ is TRUE == similarity $J(X, Y)$, provided sets X and Y are randomly choosen. So, if r is the random variable then

$h_{\min}(X) = h_{\min}(Y)$ and zero otherwise,

then r is unbiased estimator of $J(X, Y)$. r can be used as estimator for Jaccard Similarity coz it has too high variance, the value is always either zero or one. The overall overview of MinHash scheme is to reduce this variance by calculating the average of several variables constructed in the same way.

4. Stochastic Finite Automata

In computer science and mathematics, the Probabilistic Automation(P) is generalization of non-deterministic finite automaton. This also includes converting probability of any given transition into a transition function. This turns into a stochastic or transition matrix. So, the probabilistic automaton generalizes concept of sub-shift of finite type or Markov chain. Thus, these languages are known as stochastic languages.

Finite automaton A = (S, P, i, δ, T) where

P: finite input alphabet

δ : is a function: δ : S x A -> S. This is known as transition function.

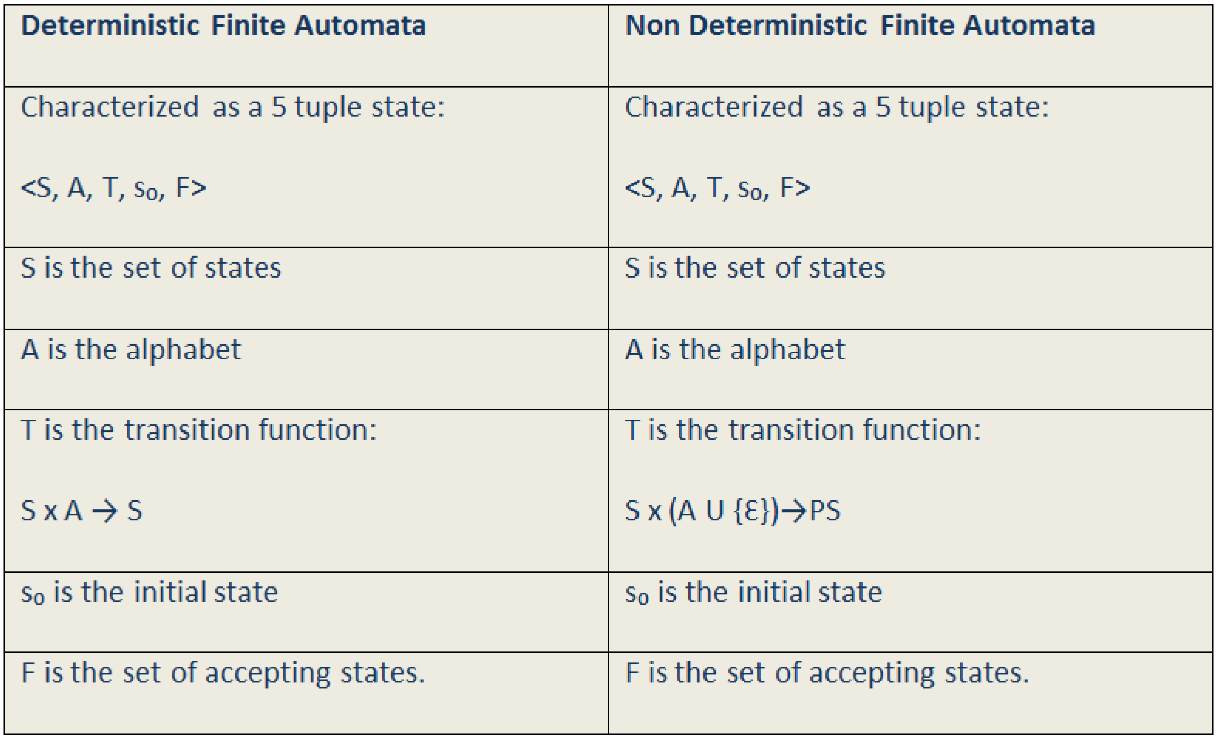
S: is a finite set known as set of states.

T: is subset of S known as terminal state.

Non-Deterministic Finite Automata(NDFA):

This type of automata has multiple acceptance state at any instance of time. Transition from one state to another state is possible in many number of states.

Deterministic Finite Automata Vs Non-Deterministic Finite Automata [13]:

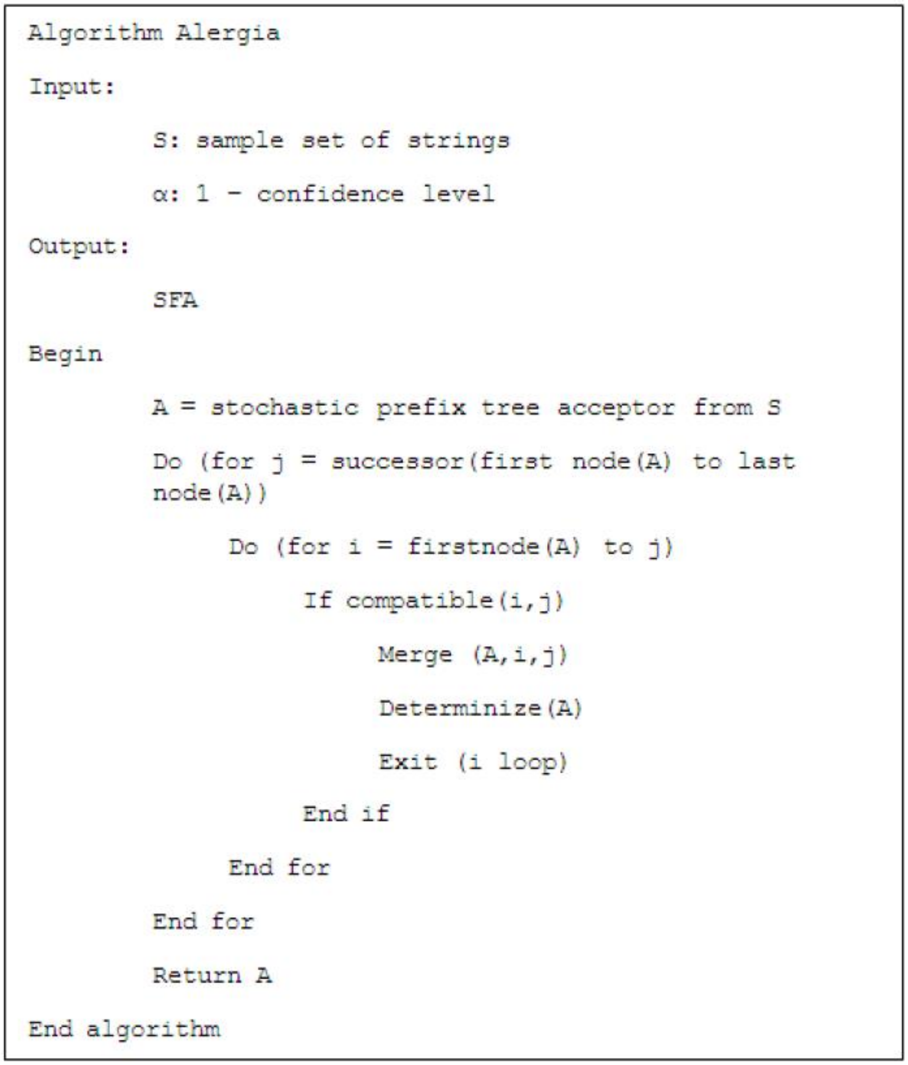


Deterministic Finite Automata	**Non Deterministic Finite Automata**
Characterized as a 5 tuple state: $<S, A, T, s_0, F>$	Characterized as a 5 tuple state: $<S, A, T, s_0, F>$
S is the set of states	S is the set of states
A is the alphabet	A is the alphabet
T is the transition function: $S \times A \rightarrow S$	T is the transition function: $S \times (A \cup \{\varepsilon\}) \rightarrow PS$
s_0 is the initial state	s_0 is the initial state
F is the set of accepting states.	F is the set of accepting states.

Table 1: Differences between Deterministic Finite Automata & Non Deterministic Finite Automata

4.1. Alergia Algorithm

Alergia Algorithm is can be determined as Stochastic Finite State Transducer (SFST) [14][15]. We use Alergia Algorithm for our author pattern recognition. Below is the algorithm.



```
Algorithm Alergia
Input:
        S: sample set of strings
        α: 1 – confidence level
Output:
        SFA
Begin
        A = stochastic prefix tree acceptor from S
        Do (for j = successor(first node(A) to last
        node(A))
             Do (for i = firstnode(A) to j)
                  If compatible(i,j)
                       Merge (A,i,j)
                       Determinize(A)
                       Exit (i loop)
                  End if
             End for
        End for
        Return A
End algorithm
```

5. Function Words

Function words [8] are words that have lexical meaning, which serves to express grammatical relationships with different words in a sentence, or specify mood or attitude of a speaker. In some cases, function words might also have ambiguous meaning.

In the phase of the project, the functional words are stop words. There are 261 function words which have been categorized into 7 different types by Stanford. These serve the purpose for differentiating among various stop words and for easy computation. Below is the list of function words with their equivalent weight.

Table 2: List of function words

Sl. No	Function Word	Category
1	a	3
2	able	1
3	aboard	4
4	about	4
5	above	4
6	absent	4
7	according	4
8	accordingly	2
9	across	4
10	after	4
11	against	4
12	ahead	4

Sl. No	Function Word	Category
131	must	1
132	my	3
133	myself	5
134	near	4
135	need	1
136	neither	3
137	nevertheless	2
138	next	4
139	no	3
140	no_one	5
141	nobody	5
142	none	5

13	albeit	2
14	all	6
15	along	4
16	alongside	4
17	although	2
18	amid	4
19	amidst	4
20	among	4
21	amongst	4
22	amount	6
23	an	3
24	and	2
25	another	3
26	anti	4
27	any	3
28	anybody	5
29	anyone	5
30	anything	5
31	around	4
32	as	2
33	aside	4
34	astraddle	4
35	astride	4

143	nor	2
144	nothing	5
145	notwithstanding	4
146	number	6
147	numbers	6
148	of	4
149	off	4
150	on	4
151	once	2
152	one	5
153	onto	4
154	opposite	4
155	or	2
156	other	3
157	ought	1
158	our	3
159	ours	5
160	ourselves	5
161	out	4
162	outside	4
163	over	4
164	part	6
165	past	4

36	at	4
37	away	4
38	bar	4
39	barring	4
40	be	1
41	am	1
42	are	1
43	is	1
44	was	1
45	were	1
46	because	2
47	before	4
48	behind	4
49	below	4
50	beneath	4
51	beside	4
52	besides	4
53	between	4
54	beyond	4
55	bit	6
56	both	3
57	but	2
58	by	4

166	pending	4
167	per	4
168	pertaining	4
169	place	4
170	plenty	6
171	plethora	6
172	plus	4
173	quantities	6
174	quantity	6
175	regarding	4
176	remainder	6
177	respecting	4
178	rest	6
179	round	4
180	save	4
181	saving	4
182	several	6
183	shall	1
184	she	5
185	should	1
186	similar	4
187	since	4
188	so	2

59	can	1
60	certain	6
61	circa	4
62	close	4
63	concerning	4
64	consequently	2
65	considering	4
66	could	1
67	couple	6
68	dare	1
69	despite	4
70	down	4
71	due	4
72	during	4
73	each	3
74	either	3
75	enough	6
76	every	3
77	everybody	5
78	everyone	5
79	everything	5
80	except	4
81	excepting	4
189	some	6
190	somebody	5
191	someone	5
192	something	5
193	spite	4
194	such	5
195	than	2
196	that	2
197	the	3
198	their	3
199	theirs	5
200	them	5
201	themselves	5
202	then	2
203	therefore	2
204	these	3
205	they	5
206	this	3
207	those	3
208	though	2
209	through	4
210	throughout	4
211	thru	4

82	excluding	4
83	failing	4
84	few	6
85	fewer	6
86	following	4
87	for	4
88	from	4
89	front	4
90	have	1
91	has	1
92	had	1
93	he	5
94	heaps	6
95	hence	2
96	her	5
97	hers	5
98	herself	5
99	him	5
100	himself	5
101	his	5
102	however	2
103	i	5
104	if	2

212	thus	2
213	till	4
214	to	4
215	tons	6
216	top	4
217	toward	4
218	towards	4
219	under	4
220	underneath	4
221	unless	2
222	unlike	4
223	until	4
224	unto	4
225	up	4
226	upon	4
227	us	5
228	various	6
229	versus	4
230	via	4
231	wanting	4
232	we	5
233	what	5
234	whatever	2

105	in	4
106	including	4
107	inside	4
108	instead	4
109	into	4
110	it	5
111	its	5
112	itself	5
113	lack	6
114	less	6
115	like	4
116	little	6
117	loads	6
118	lots	6
119	majority	6
120	many	6
121	masses	6
122	may	1
123	me	5
124	might	1
125	mine	5
126	minority	6
127	minus	4
235	when	2
236	whenever	2
237	where	2
238	whereas	2
239	wherever	2
240	whether	2
241	which	5
242	whichever	2
243	while	2
244	whilst	2
245	who	5
246	whoever	2
247	whole	6
248	whom	5
249	whomever	2
250	whose	5
251	will	1
252	with	4
253	within	4
254	without	4
255	would	1
256	yet	2
257	you	5

128	more	6
129	most	6
130	much	6
258	your	3
259	yours	5
260	yourself	5
261	yourselves	5

Below are the sentences with the application of function words

Examples:

Sentence: "**Function words [8] are words that have lexical meaning, which serves to express grammatical relationships with different words in a sentence, or specify mood or attitude of a speaker**"

Function word sentence: are that have little or have but instead to with other within a

Word category: 1 2 1 6 2 1 2 4 4 4 3 4 1 2 3 2 4 3

Now the complete sentence has been converted to list of integers 1 2 1 6 2 1 2 4 4 4 3 4 1 2 3 2 4 3. This list is used to identify authors pattern.

6. Dataset & Environment details

6.1 Dataset details

The dataset I have chosen for this project is Novels written by different authors. These novels are available on internet and can be obtained for free for educational purpose. The size of the document gives a cutting edge to form clusters authors. Below are few numerical details about the dataset.

- Avg. no of lines in a document: ~9k
- Number of documents: 33
- Number of authors: 27

6.2. Environment details

- Processor: i7 4th Gen 2.40 Ghz
- RAM: 8GB
- Operating System: Windows 10
- Programming language: JAVA 64-bit
- Graphical Memory: 2GB Intel HD 4700 Graphics, 2GB NVIDIA 655M

7. Training & Test datasets

Documents considered in the for training the model.

1. Adam Smith - An Inquiry into the nature.txt
2. Alexandre Dumas - The Count of Monte Cristo.txt
3. Arthur Conan Doyle -The Adventures of Sherlock Holmes.txt
4. Bram Stoker - Dracula.txt
5. Canan Doyle - A Study in Scarlet.txt
6. Charles Dickens - David Copperfield.txt
7. Charlotte Bronte - Jane Eyre.txt
8. Daniel Defoe - Robinson Crusoe.txt
9. Dante Alighieri - The Divine Comedy.txt
10. Edgar Rice Burroughs - A Princess of Mars.txt
11. Elliott Whithey - The Pirate Shark.txt
12. Frank Baum - The Wonderful Wizard of Oz.txt
13. Friedrich Nietzsche - Beyond Good and Evil.txt
14. Gabriel Garcia Marquez - Years of Solitude.txt
15. Harrison Williams - Legends of Loudoun.txt
16. J.K. Rowling - Harry Potter And the Order OF Phoenix.txt
17. J.K. Rowling - Harry Potter and Sorcerer's Stone.txt
18. Herbert George Wells - The War of the Worlds.txt
19. Herman Melville - Moby Dick.txt
20. Herman See;y - A Son of the City.txt

21. Hermann Hesse- Siddhartha.txt

22. James Joyce - Dubliners.txt

23. James Matthew Barrie - Peter Pan.txt

24. Jane Austen - Emma.txt

25. John Milton - Paradise Lost.txt

26. Jonathan Swift - Gulivers Traveles.txt

27. Joseph Conrad - Heart of Darkness.txt

28. Jules Verne - Around the World in Days.txt

29. Julia Ward Howe - From the Oak to the Olive.txt

30. Leo Tolstoy - War and Peace.txt

Test document:

1. J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

8. Applying Jaccard Similarity for author recognition

Considering two documents A and B documents written by same author. All the documents have different statements. Each document will have its statements. Below are the diagrams representing documents A and B.

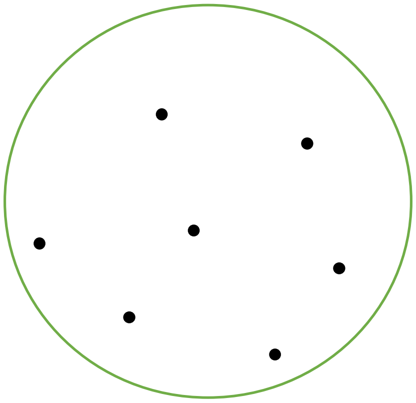


Fig 1: Document A

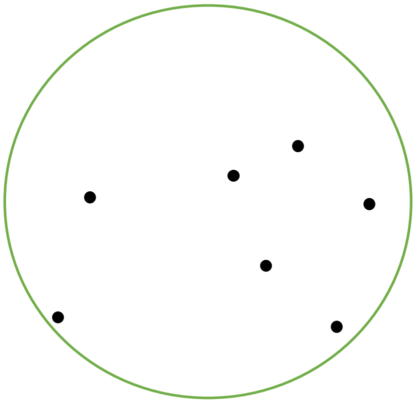


Fig 2: Document B

The dots in both the documents represent statements in the document. This statements are represented by functional words. A data pre-processing step is involved before generating document trees. After this, the documents are verified to check for common statements among the documents. Below is the figure that represents common statements between different documents.

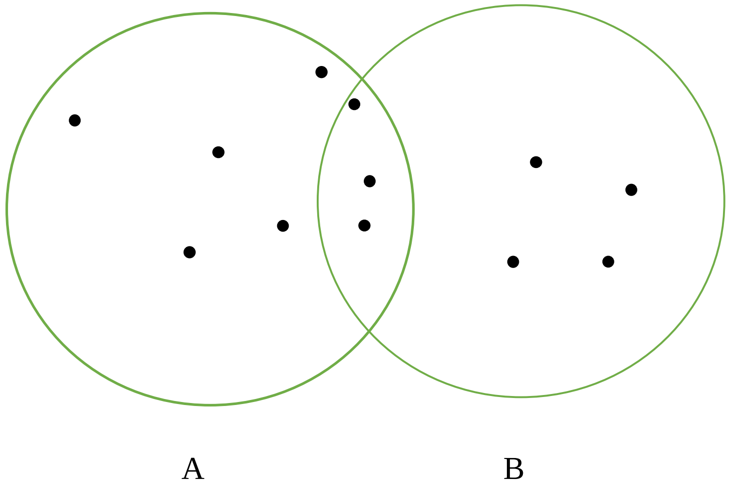


Fig 3: Identifying common statements

Total number of common statements (A ∩ B) = 3

Total number of statement (A U B) = 13

Jaccard Similarity = (A ∩ B) / (A U B) = 3 / 13.

Now consider we have added a third document from same author.

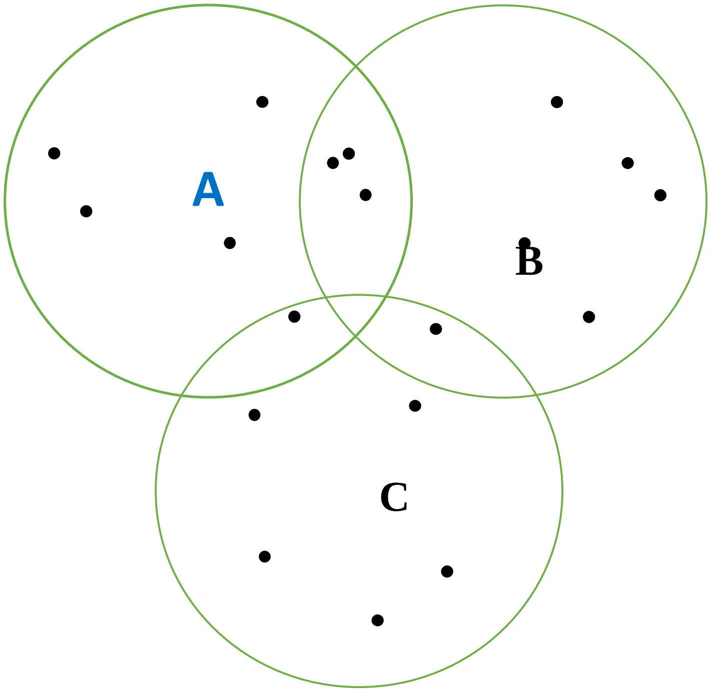


Fig 4: Set of three documents

Total number of statements in documents = (A U B U C) = 17

Total number of common statements = (A ∩ B) + (B ∩ C) + (A ∩ C) + (A ∩ B ∩ C) = 6

Jaccard Similarity = (A ∩ B) + (B ∩ C) + (A ∩ C) + (A ∩ B ∩ C) / (A U B U C) = 6 / 17

For improving the results, I have removed common statement in all the documents to improve the efficiency of the results. I did various trial to identify the minimum number of words in a statement. In the next page there are results for various variations in the minhash algorithm.

8.1. Results

Test Case 1: In this test case, we consider sentence as a sentence. This sentence is converted to function word category and Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl. No	Document name	% Match
1	Adam Smith - An Inquiry into the nature.txt	8.212237094
2	Alexandre Dumas - The Count of Monte Cristo.txt	7.661097852
3	Arthur Conan Doyle -The Adventures of Sherlock Holmes.txt	17.06359592
4	Bram Stoker - Dracula.txt	13.35579993
5	Canan Doyle - A Study in Scarlet.txt	20.48085485
6	Charles Dickens - David Copperfield.txt	9.639418152
7	Charlotte Bronte - Jane Eyre.txt	14.95482205
8	Daniel Defoe - Robinson Crusoe.txt	33.22515213
9	Dante Alighieri - The Divine Comedy.txt	16.99772985
10	Edgar Rice Burroughs - A Princess of Mars.txt	25.64102564
11	Elliott Whithey - The Pirate Shark.txt	17.95523906
12	Frank Baum - The Wonderful Wizard of Oz.txt	21.32564841
13	Friedrich Nietzsche - Beyond Good and Evil.txt	32.5433526
14	Gabriel Garcia Marquez - 100 Years of Solitude.txt	18.4086629
15	Harrison Williams - Legends of Loudoun.txt	21.67042889
16	harrypotter1.txt	100
17	harrypotter2.txt	17.28748806
18	Herbert George Wells - The War of the Worlds.txt	20.88959492
19	Herman Melville - Moby Dick.txt	14.75980931
20	Herman See;y - A Son of the City.txt	18.58648965
21	Hermann Hesse- Siddhartha.txt	21.72264355
22	James Joyce - Dubliners.txt	20.17746914
23	James Matthew Barrie - Peter Pan.txt	22.35294118
24	Jane Austen - Emma.txt	15.22994652
25	John Milton - Paradise Lost.txt	26.67380825
26	Jonathan Swift - Gulivers Traveles.txt	26.55977693
27	Joseph Conrad - Heart of Darkness.txt	22.29199372
28	Jules Verne - Around the World in 80 Days.txt	20.78720787
29	Julia Ward Howe - From the Oak to the Olive.txt	16.00249066
30	Leo Tolstoy - War and Peace.txt	6.379377685

Table 3: Results for Jaccard Similarity for Test Case 1

Test Case 2: In this test case, we consider a paragraph as a sentence. The word of the sentence is considered as symbol and Alergia Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl. No	Document name	% Match
1	Adam Smith - An Inquiry into the nature.txt	0
2	Alexandre Dumas - The Count of Monte Cristo.txt	0.014160295
3	Arthur Conan Doyle -The Adventures of Sherlock Holmes.txt	0.039936102
4	Bram Stoker - Dracula.txt	0
5	Canan Doyle - A Study in Scarlet.txt	0
6	Charles Dickens - David Copperfield.txt	0
7	Charlotte Bronte - Jane Eyre.txt	0
8	Daniel Defoe - Robinson Crusoe.txt	0
9	Dante Alighieri - The Divine Comedy.txt	0
10	Edgar Rice Burroughs - A Princess of Mars.txt	0
11	Elliott Whithey - The Pirate Shark.txt	0
12	Frank Baum - The Wonderful Wizard of Oz.txt	0.054945055
13	Friedrich Nietzsche - Beyond Good and Evil.txt	0
14	Gabriel Garcia Marquez - 100 Years of Solitude.txt	0.061050061
15	Harrison Williams - Legends of Loudoun.txt	0
16	harrypotter1.txt	100
17	harrypotter2.txt	0.116618076
18	Herbert George Wells - The War of the Worlds.txt	0
19	Herman Melville - Moby Dick.txt	0
20	Herman See;y - A Son of the City.txt	0
21	Hermann Hesse- Siddhartha.txt	0
22	James Joyce - Dubliners.txt	0.049578582
23	James Matthew Barrie - Peter Pan.txt	0
24	Jane Austen - Emma.txt	0
25	John Milton - Paradise Lost.txt	0
26	Jonathan Swift - Gulivers Traveles.txt	0
27	Joseph Conrad - Heart of Darkness.txt	0
28	Jules Verne - Around the World in 80 Days.txt	0
29	Julia Ward Howe - From the Oak to the Olive.txt	0
30	Leo Tolstoy - War and Peace.txt	0

Table 4: Results for Jaccard Similarity for Test Case 2

Test Case 3: In this test case, we consider a paragraph as a sentence. These sentences are converted to functional word categories and Alergia Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl. No	Document name	% Match
1	Adam Smith - An Inquiry into the nature.txt	34.1888176
2	Alexandre Dumas - The Count of Monte Cristo.txt	15.31633617
3	Arthur Conan Doyle -The Adventures of Sherlock Holmes.txt	47.65517241
4	Bram Stoker - Dracula.txt	46.69163546
5	Canan Doyle - A Study in Scarlet.txt	37.0332997
6	Charles Dickens - David Copperfield.txt	26.54664484
7	Charlotte Bronte - Jane Eyre.txt	32.60120586
8	Daniel Defoe - Robinson Crusoe.txt	65
9	Dante Alighieri - The Divine Comedy.txt	53.90702275
10	Edgar Rice Burroughs - A Princess of Mars.txt	74.87487487
11	Elliott Whithey - The Pirate Shark.txt	40.75587334
12	Frank Baum - The Wonderful Wizard of Oz.txt	43.04029304
13	Friedrich Nietzsche - Beyond Good and Evil.txt	16.66666667
14	Gabriel Garcia Marquez - 100 Years of Solitude.txt	54.81727575
15	Harrison Williams - Legends of Loudoun.txt	49.94571118
16	harrypotter1.txt	100
17	harrypotter2.txt	31.45961266
18	Herbert George Wells - The War of the Worlds.txt	56.8788501
19	Herman Melville - Moby Dick.txt	43.07159353
20	Herman See;y - A Son of the City.txt	41.60331721
21	Hermann Hesse- Siddhartha.txt	28.86266094
22	James Joyce - Dubliners.txt	50.77343039
23	James Matthew Barrie - Peter Pan.txt	40.97807757
24	Jane Austen - Emma.txt	40.21864212
25	John Milton - Paradise Lost.txt	26.53061224
26	Jonathan Swift - Gulivers Traveles.txt	47.67063922
27	Joseph Conrad - Heart of Darkness.txt	17.28395062
28	Jules Verne - Around the World in 80 Days.txt	54.66786355
29	Julia Ward Howe - From the Oak to the Olive.txt	39.51434879
30	Leo Tolstoy - War and Peace.txt	11.36458478

Table 5: Results for Jaccard Similarity for Test Case 3

Conclusion

From the above results it can be observed that Jaccard Similarity doesn’t give good matching results. In all the above 3 test cases, the results were negligible and doesn’t follow any good pattern to come to proper conclusion. We can conclude that this method can’t be used to identify the author of the document.

9. Applying Stochastic Finite Automata using Alergia Algorithm to recognize author

Below are the experimental results for a variety of Alpha values ranging from 0.1 to 0.95 by incrementing the value of alpha by 0.05

Test Case 1: In this test case, we consider sentence as a sentence. This sentence is converted to function word category and Alergia Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl.No	Alpha	Doc01	Doc02	Doc03	Doc04	Doc05	Doc06	Doc07	Doc08	Doc9	Doc10	Doc11	Doc12	Doc13	Doc14	Doc15
1	0.1	99.791	99.834	99.669	99.811	99.646	99.789	99.814	99.182	99.776	99.62	99.892	99.911	99.632	99.846	99.918
2	0.15	91.378	95.587	96.856	97.227	96.743	96.692	95.974	82.74	95.842	90.702	98.886	97.294	93.33	93.251	96.2
3	0.2	79.873	92.254	93.229	94.384	93.168	94.293	92.468	73.783	91.326	86.855	94.682	91.571	81.933	87.89	89.311
4	0.25	78.488	92.642	93.629	93.41	93.097	92.773	91.439	66.38	91.482	84.108	96.083	92.236	82.458	86.348	91.006
5	0.3	74.839	91.001	92.857	91.442	92.425	92.281	90.502	67.239	85.245	82.164	94.43	90.284	77.994	86.83	88.218
6	0.35	77.652	93.499	94.098	93.718	93.947	93.608	93.331	68.262	91.08	86.56	96.299	91.659	82.09	87.91	90.623
7	0.4	59.981	86.459	88.555	86.443	89.062	85.318	84.936	40.532	77.867	67.16	91.197	83.718	65.389	68.415	76.545
8	0.45	44.896	78.746	80.323	75.42	78.372	77.186	76.078	30.92	67.27	54.522	86.238	74.889	54.307	57.809	70.886
9	0.5	39.052	75.164	78.737	74.903	76.354	75.384	72.35	27.853	61.569	48.648	85.411	70.763	50.945	54.435	66.594
10	0.55	41.7	78.461	81.509	78.442	79.788	78.82	76.885	31.943	66.644	53.931	86.921	73.957	54.044	59.969	68.972
11	0.6	32.834	72.112	74.214	70.41	71.186	72.7	69.511	23.845	56.919	43.068	80.704	66.327	45.693	49.73	62.575
12	0.65	29.498	71.188	73.525	68.84	70.372	71.852	69.149	24.049	56.003	41.082	79.806	66.105	45.221	47.532	60.361
13	0.7	23.306	66.439	69.719	63.373	65.062	67.916	64.196	21.391	48.849	34.784	75.422	58.917	40.599	42.287	56.561
14	0.75	17.07	58.678	61.473	53.394	55.823	61.242	55.858	16.933	39.347	25.106	69.206	50.665	34.559	35.017	51.258
15	0.8	16.713	58.238	59.68	51.526	53.77	60.061	56.145	16.728	38.006	23.838	67.481	48.625	35.137	33.378	50.273
16	0.85	15.137	56.763	58.977	50.184	52.425	59.085	54.244	15.91	36.15	24.26	66.439	45.963	32.878	32.356	49.563
17	0.9	13.369	54.032	55.736	46.804	50.867	56.766	52.045	15.91	32.238	21.682	63.852	42.325	32.353	29.136	47.048
18	0.95	12.393	50.972	52.592	44.121	46.938	54.433	48.817	14.233	30.025	19.949	60.87	40.151	30.252	27.902	45.927

Sl.No	Alpha	Doc16	Doc17	Doc18	Doc19	Doc20	Doc21	Doc22	Doc23	Doc24	Doc25	Doc26	Doc27	Doc28	Doc29	Doc30
1	0.1	100	99.882	99.91	99.806	99.949	99.839	99.908	99.789	99.77	99.688	99.688	99.842	99.901	99.954	99.851
2	0.15	100	99.197	97.164	95.787	98.443	94.675	98.384	98.193	97.969	92.15	87.296	97.825	97.618	95.861	97.194
3	0.2	100	97.59	91.071	90.382	96.766	89.403	96.896	94.096	93.85	85.483	81.291	93.278	93.5	90.22	93.436
4	0.25	100	97.815	93.575	91.127	97.091	89.779	96.381	94.639	92.26	83.614	76.362	93.99	95.014	92.277	93.767
5	0.3	100	96.841	91.976	87.176	95.756	85.745	95.463	94.036	91.436	73.956	75.668	92.962	93.203	88.809	91.984
6	0.35	100	97.751	92.881	90.74	96.629	91.555	96.675	94.217	92.633	82.617	79.104	93.99	95.361	92.671	93.974
7	0.4	100	95.277	83.71	80.744	93.515	81.119	93.148	89.337	87.125	53.707	56.022	87.9	88.787	86.15	85.971
8	0.45	100	91.315	76.893	72.888	89.973	71.867	86.976	81.386	79.184	43.863	44.186	80.467	82.039	77.896	78.271
9	0.5	100	90.587	74.811	70.002	87.953	66.541	86.315	79.006	76.32	42.555	39.327	79.992	79.558	74.243	76.464
10	0.55	100	92.086	75.716	71.658	89.63	72.727	88.134	82.169	79.174	45.296	43.457	81.257	83.032	75.861	79.693
11	0.6	100	87.717	68.477	65.798	85.489	63.206	83.229	76.024	73.455	37.009	35.127	75.603	75.862	68.231	72.38
12	0.65	100	87.92	67.722	64.8	84.326	61.592	83.321	76.476	73.273	35.576	30.719	74.614	74.671	67.145	71.573
13	0.7	100	84.547	61.719	59.58	81.828	56.59	78.325	71.265	68.426	29.533	26.449	70.621	69.685	58.497	67.085
14	0.75	100	78.786	51.855	53.845	75.205	47.768	70.757	63.735	61.165	26.044	20.132	61.17	62.516	49.225	59.513
15	0.8	100	77.918	50.498	52.208	74.196	45.939	70.261	62.259	59.747	23.738	19.715	61.329	61.871	47.376	58.454
16	0.85	100	77.318	50.226	51.327	73.101	44.701	69.453	61.175	58.291	23.676	19.16	59.668	60.705	45.087	57.133
17	0.9	100	73.463	45.52	49.429	69.713	41.474	65.209	58.554	55.925	22.555	16.696	56.781	56.462	40.486	54.002
18	0.95	100	71.375	44.585	47.365	68.036	39.699	63.281	55.663	53.214	20.249	15.238	53.341	55.296	38.89	51.592

Table 6: Representing Alergia results for Test Case 1

Test Case 2: In this test case, we consider a paragraph as a sentence. The word of the sentence is considered as symbol and Alergia Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl.No	Alpha	Doc01	Doc02	Doc03	Doc04	Doc05	Doc06	Doc07	Doc08	Doc9	Doc10	Doc11	Doc12	Doc13	Doc14	Doc15
1	0.1	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
2	0.15	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
3	0.2	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
4	0.25	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
5	0.3	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
6	0.35	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
7	0.4	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
8	0.45	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
9	0.5	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
10	0.55	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
11	0.6	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
12	0.65	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
13	0.7	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
14	0.75	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
15	0.8	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
16	0.85	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
17	0.9	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173
18	0.95	0.048	0	0	0	0	0	0	0.157	0.145	0.114	0.186	0	0.426	0	0.173

Sl.No	Alpha	Doc16	Doc17	Doc18	Doc19	Doc20	Doc21	Doc22	Doc23	Doc24	Doc25	Doc26	Doc27	Doc28
1	0.1	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
2	0.15	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
3	0.2	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
4	0.25	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
5	0.3	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
6	0.35	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
7	0.4	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
8	0.45	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
9	0.5	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
10	0.55	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
11	0.6	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
12	0.65	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
13	0.7	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
14	0.75	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
15	0.8	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
16	0.85	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
17	0.9	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013
18	0.95	0	0.059	0	0.295	0.11	0	0	0	0.204	0.625	0	0	0.013

Table 7: Representing Alergia results for Test Case 2

Test Case 3: In this test case, we consider a paragraph as a sentence. These sentences are converted to functional word categories and Alergia Algorithm is applied. Below are the results of comparison.

Test Document: J.K. Rowling - Harry Potter and the Order OF Phoenix.txt

Sl.No	Alpha	Doc01	Doc02	Doc03	Doc04	Doc05	Doc06	Doc07	Doc08	Doc9	Doc10	Doc11	Doc12	Doc13	Doc14	Doc15
1	0.1	99.519	99.754	99.719	99.605	99.607	99.514	99.793	99.215	98.547	99.657	99.628	100	98.298	99.426	99.655
2	0.15	97.448	98.492	98.172	95.455	95.874	96.079	98.263	91.523	96.948	98.398	98.138	98.584	96.17	94.073	98.791
3	0.2	77.805	96.165	95.077	90.711	92.731	89.663	94.582	82.575	87.936	95.195	95.903	96.884	81.702	83.365	90.674
4	0.25	57.342	90.118	87.693	84.321	87.23	79.909	87.676	63.265	79.215	88.558	90.317	93.768	65.957	64.054	74.266
5	0.3	41.839	75.418	69.409	58.564	67.191	55.412	70.389	32.339	56.977	71.167	80.074	74.079	55.319	39.197	64.076
6	0.35	11.796	42.822	33.966	20.619	32.22	17.272	34.326	8.006	22.384	21.854	38.92	37.394	33.617	9.751	25.734
7	0.4	14.636	53.458	47.257	31.555	41.847	26.539	44.789	10.832	25.872	40.847	45.81	51.133	40.426	15.679	34.37
8	0.45	3.274	26.811	21.73	11.067	13.556	7.745	21.05	1.256	7.849	5.95	15.829	14.448	20.426	4.015	17.271
9	0.5	9.677	41.396	35.865	19.697	30.452	16.008	34.202	5.181	19.477	20.595	34.637	36.261	34.043	9.369	26.598
10	0.55	6.211	32.825	27.496	11.989	21.415	9.754	25.31	3.297	12.064	11.67	22.16	26.771	28.085	6.501	20.035
11	0.6	2.215	20.01	15.893	5.599	8.448	5.476	13.275	0.942	4.651	3.89	12.477	10.765	15.319	3.442	14.162
12	0.65	4.574	30.121	24.473	11.331	18.468	10.078	23.863	2.669	9.884	10.297	18.994	23.654	20.426	5.354	17.789
13	0.7	2.696	18.928	14.909	4.809	11.395	4.439	12.283	0.628	4.215	3.318	11.546	9.49	14.043	2.486	13.299

14	0.75	2.215	21.712	15.208	11.603	3.755	8.251	3.986	10.008	0.628	3.779	2.746	8.194	7.082	10.213	2.486	13.817
15	0.8	2.311	23.222	16.47	13.08	4.809	9.43	4.861	11.828	0.942	4.506	4.233	11.173	9.065	11.489	3.059	15.026
16	0.85	1.926	19.132	13.389	10.408	3.821	5.894	3.467	8.809	0.628	2.762	3.089	9.125	6.374	10.638	1.53	13.99
17	0.9	0.818	13.342	7.932	6.048	2.24	4.912	2.171	5.418	0.628	1.744	1.487	3.724	3.258	4.255	1.53	11.917
18	0.95	1.348	13.531	8.538	5.696	2.569	3.929	2.041	5.376	0.628	1.308	0.572	4.283	3.399	5.532	1.912	11.917

Sl.No	Alpha	Doc16	Doc17	Doc18	Doc19	Doc20	Doc21	Doc22	Doc23	Doc24	Doc25	Doc26	Doc27	Doc28	Doc29	Doc30
1	0.1	100	99.914	99.851	99.524	99.852	99.705	99.45	99.79	99.814	99.353	98.569	98.125	100	99.482	99.796
2	0.15	100	99.183	99.552	97.741	99.113	97.345	98.02	98.952	97.585	94.175	91.82	93.75	98.762	96.891	98.739
3	0.2	100	98.023	95.224	87.277	97.044	92.625	94.169	98.008	92.26	74.434	77.505	81.875	96.078	83.679	95.071
4	0.25	100	95.316	86.269	78.478	93.422	83.776	90.429	92.977	84.087	70.55	63.804	66.875	88.545	69.948	89.786
5	0.3	100	84.186	68.06	61.534	84.479	63.422	72.607	79.14	59.133	49.191	40.082	43.75	78.844	43.523	76.159
6	0.35	100	59.132	27.164	24.732	53.511	23.599	44.994	51.992	23.777	21.359	12.679	9.375	43.963	10.104	42.091
7	0.4	100	65.234	42.388	31.748	62.158	36.578	49.835	60.168	32.941	21.036	19.632	20.625	57.069	14.249	52.292
8	0.45	100	34.207	10	10.523	26.755	9.44	27.393	34.172	12.817	9.709	3.885	1.875	24.149	3.886	20.899
9	0.5	100	55.952	24.03	23.722	49.446	21.239	41.694	50.629	24.272	14.563	11.043	10.625	43.653	8.549	40.448
10	0.55	100	42.931	15.075	15.101	35.255	17.109	33.333	40.042	16.471	12.298	7.975	3.125	31.992	4.922	29.215
11	0.6	100	25.698	6.119	7.313	20.547	9.44	21.012	26.31	6.687	6.149	3.067	1.875	18.473	2.332	15.27
12	0.65	100	39.837	14.478	13.139	35.255	13.274	32.013	36.268	13.251	8.091	6.953	3.125	29.205	4.922	26.707
13	0.7	100	23.893	6.269	6.659	18.477	8.26	18.702	22.956	6.006	4.531	3.272	1.875	19.917	2.85	13.869
14	0.75	100	19.424	6.269	5.945	16.334	8.555	15.512	17.82	5.201	5.825	3.272	0.625	14.861	2.85	11.768
15	0.8	100	22.303	7.164	7.372	18.551	9.145	16.942	21.384	5.882	5.825	3.681	0.625	14.964	2.591	13.029
16	0.85	100	18.049	4.925	5.351	14.265	4.72	14.411	18.658	4.334	5.502	1.431	1.25	13.209	2.85	10.596
17	0.9	100	10.786	2.687	3.805	8.278	4.13	8.691	10.901	1.981	3.56	2.045	0.625	8.772	2.073	5.489
18	0.95	100	11.818	2.985	3.032	9.387	3.54	9.351	10.692	2.477	3.56	1.636	0.625	7.637	1.554	5.884

Table 8: Representing Alergia results for Test Case 3

Conclusion:

From the above test cases, we can say that Test case 2 doesn’t make much sense for converting whole sentence into a paragraph and considering all the possible words. But Test case 1 and 3 does make sense as we shorten the paragraph by function words and which gives good results in identifying documents written by similar author.

10. Future Work

I would like to test this in cloud environment implementing the concept of computing grids or in big data environment. Also including various text processing techniques like stemming and lemmatization. I would also like to test this application with other algorithms like Alergia. This application might also be helpful to identify plagiarism.

11. Conclusion

Automata with Alergia builds a transition tree which represents the writing pattern of an author. This mechanism helps to identify the similarity between the documents. Based on this similarity we can determine the original author of a document with the % match. If the document size is too small the results may not be accurate but, as the document size increases there is high possibility that automata will give accurate result. Furthermore, this algorithm also helps to determine plagiarized content of a document. Using this algorithm, we were able to determine that harry potter author was recognized and the % match with the document is high. So, use of Alergia algorithm does a good job of recognizing author of the document by developing the author writing pattern.

Where as in the experiment 2 using Jaccard similarity, which doesn't develop author writing pattern, but it does a good job of calculating distance between sentences. According to the initial ideology and assumptions, this result was supposed to show good match with similarity author. But, the results where negative based on the calculating distance between the sentences.

So, based on the above conclusion we can derive that experiment 2 using Jaccard Similarity can't be used for recognizing authors pattern and Automata with Alergia can be used for recognizing similarity between the documents and author.

12. References

1. Slaney, M ; Yahoo! Research, Sunnyvale, CA , USA ; Lifshits, Y. ; Junfeng He “Optimal Parameters for Locality-Sensitive Hashing 2012”
2. Locality Sensitive Hashing: https://en.wikipedia.org/wiki/Locality-sensitive_hashing
3. Oprisa, C. "A MinHash Approach for Clustering Large Collections of Binary Programs", *Control Systems and Computer Science (CSCS), 2015 20th International Conference on,* On page(s): 157 – 163
4. Oprisa, C.; Checiches, M.; Nandrean, A. "Locality-sensitive hashing optimizations for fast malware clustering", *Intelligent Computer Communication and Processing (ICCP), 2014 IEEE International Conference on,* On page(s): 97 – 104
5. Rajaraman, A.; Ullman, J. (2010). "Mining of Massive Datasets, Ch. 3.".
6. Gionis, A.; Indyk, P.; Motwani, R. (1999). "Similarity Search in High Dimensions via Hashing". Proceedings of the 25th Very Large Database (VLDB) Conference.
7. Indyk, Piotr.; Motwani, Rajeev. (1998). "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality.". Proceedings of 30th Symposium on Theory of Computing.
8. Function Words : https://en.wikipedia.org/wiki/Function_word
9. Wikipedia: https://en.wikipedia.org/wiki/Wikipedia
10. Wikipedia Dataset: http://wiki.dbpedia.org/Datasets
11. Josh Levs, CNN: J.K. Rowling revealed as secret author of crime novel, http://www.cnn.com/2013/07/14/world/rowling-secret-book/
12. Hamming Distance: https://en.wikipedia.org/wiki/Hamming_distance
13. Rajeev Motwani, Jeffrey D. Ullman, John E. Hopcroft.: Introduction to Automata Theory, Languages, And Computation (2003)

14. Jir Poner, Filip Lanka.: Computational Studies of RNA and DNA. Challenges and Advances in Computational Chemestry and Physics (2006)

15. Ferdinand Wagner, Ruedi Schmuki, Thomas Wagner, Peter Wolstenholme.: Modeling Software with Finite State Machines: A Practical Approach (2006)

16. Min Hashing: https://en.wikipedia.org/wiki/MinHash