

Plagiarism Detection based on studying correlation between Author, Title and Content

Abeer Al Jarrah
Yarmouk University
Faculty of Computer and
Information Technology
aljarraha@cbj.gov.jo

Dr.Izzat Alsmadi
Yarmouk University
Faculty of Computer and
Information Technology
alsmadi@yu.edu.jo

Dr.Zakariya Za'atreh
Yarmouk University
Faculty of Computer and
Information Technology
zzaatreh@yu.edu.jo

Abstract

Recently, the problem of plagiarism is becoming an important issue in many debates in the fields of Education and Technology. The wide use and availability of electronic resources makes it easy for students, authors and even academic people to access and use any piece of information and embed it into his/ her own work without proper citation. The problem is raising in an exponential manner the thing which puts the education process under threat.

Several tools are presented to solve the problem of automating plagiarism detection each of which has its own good and bad features, but still the traditional way of plagiarism detection through free text search using search engines is considered an accurate and free way to detect plagiarism with the only disadvantage of being a time consuming method.

This research intends to present an alternative to plagiarism detection tools by automating the traditional free search process on search engines to detect plagiarism by intelligently extracting selective parts of text from the file subject to check and pass them to search engine in different forms and processing results in order to come up with a decision of committing plagiarism in a certain degree.

The approach used in this paper is to make string comparison of the text with the global *www*, which makes it more comprehensive compared to other plagiarism tools that depend on specific databases.

Key words: Plagiarism Detection, Automated search, correlation between Author ,Title and Content.

1. Introduction

1.1.Research Background

The widespread use of electronic resources in our daily life and the easiness of

extracting, modifying and using data in many electronic formats have imposed a lot of opportunities as well as challenges. The limitless knowledge that is offered through the internet and other electronic resources initially aimed to provide information to

knowledge seekers anytime-anywhere, and it succeed in that in a remarkable way but unfortunately led to other serious problems that may hardly and directly affect the knowledge quality itself. Plagiarism is a serious phenomenon that is spreading widely in all levels of academia as well as in all sorts of intellectual work.

Plagiarism or Digital Plagiarism as mentioned by Butakov. & Scherbinin. (2009), has many forms and therefore many definitions, according to Alzahrani et al. (2009) it may include copying part or all of the text and use it without referring it to the original composer, rephrasing the text by changing the words used but expressing the same ideas from others work, translating others work from language to language without referencing the right references and source code cloning which is done by using any piece of programming source code developed by other programmers and reuse without proper citation or even permission to.

Education expert Ryan. (2007) has discussed how serious the problem of plagiarism is, and how it seriously produce poor academic performance; because simply keeping the quality of learning will not be guaranteed if we couldn't maintain the quality of all learning processes including assessment process which is one of the most important processes that is hardly affected by plagiarism.

The solution to plagiarism can be maintained by following two approaches, plagiarism prevention and plagiarism detection. Alzahrani et al. (2009) explained that **Prevention** will be achieved by providing the ability of **detecting** the unoriginal content of student assignments or any kind of work thus affecting judging and assessing that work. Our project intents to provide a tool for plagiarism detection by

automating a robot whose mission is to uncover plagiarism by detecting the parts plagiarized and defining a percentage of plagiarism for each part.

The trend of automating plagiarism detection is becoming more and more severe as the volume and intensity of plagiarism is increasing. (Lukashenko et al., 2007)

1.2. Statement of Problem

“My sense is that Internet plagiarism is becoming more dangerous than we realize.”

Ellen Laird

According to research the percentage of plagiarism is increasing rapidly in Universities and in companies to a limit that we should pay attention to this disease, one study by Butakov. & Scherbinin. (2009) showed that more than 90% of students in high schools plagiarize as a common practice; more than 10% of students in Universities plagiarize in the USA, Australia and UK even in MIT, more that 30% of students were punished due to conduct of plagiarism (Ji. et al., 2008). These large percentages should switch the alarm to better discover tools to prevent and detect this act of dishonesty that having popularity among students of the modern age.

Plagiarism in American schools is becoming more serious over time; it is often described as an epidemic, according to some studies plagiarism at the University of California at Berkley increased by 744% from 1993 to 1997. (Lukashenko., 2007)

Online learning should insure that course design and assessment of students according to academic honesty standards that not less than that required in traditional learning.

Different ways can be used to check for plagiarism starting from manually using search engines and passing quoted sentences and search engine will return pages containing these sentences or if user wants more complicated and automated detection user can use a set of well known programs

like IntegriGuard through PaperBin.com, HowOriginal.com and Plagiarism.org .

According to Total Quality Management principles, to insure quality of learning process all phases of learning - including assessment - should comply to quality standards, and to achieve this specially when dealing with challenges of Online-Education, assessment should be controlled by automated tools for plagiarism checking and detection.(Rovai., 2000)

For Arab countries, the challenge is even harder, since the number of students is constantly rising and the governmental goals move in the direction of offering education to all, wither using the traditional learning environments or moving to the contemporary E-Learning system. Thus it would be nearly impossible for instructors to spend enough time to grade assignments and to evaluate students thoroughly enough to discover plagiarism in their work.

The call for Automatic Plagiarism Detection Tools is not new, many algorithms and tools were designed and used each of which has its own strengths and weaknesses that will discussed later in our literature.

1.3. Research Objectives

Our Research aims to:

- To implement Algorithm that can classify plagiarism into degrees according to plagiarism threshold

determined by the experimental studies of our project.

- To have the ability to access any document published on the web and to compare it with the document subject to test for plagiarism.
- To have efficient performance (search time) relative to other existent tools.

1.4. Project Scope

The Project in its current situation will use only Title, Authors and Keywords for detection plagiarism as many articles directories full text may not be available to search engines but their Meta data like Title, Author and some keywords are available which enables detecting these resources. This methodology makes our project ideal for journals and organizations that accepts academic papers and needs to check if they are already published before or if the same author has published the paper in different title or if the same title is published with different author/s, it will not cover body text plagiarism because it is not using full text in the search process.

1.5. Tools and Implementation

The tools used are mainly offered by .Net libraries, the programming language used to implement this project is C# through the use of Microsoft Visual Studio.Net 2008; many libraries were used to help perform different tasks demonstrated in Table-1.

Library Name	Purpose of Use
Watin.Core	Automate the Use of Internet Explorer
EPocalipse.IFilter	Extract Text from PDF Documents
Microsoft.Office.Interop.Word	Extract Text from Word Documents
System.Text.RegularExpressions	Extract Text from HTML Pages

Table-1: External libraries used

2. Literature Review

2.1. Plagiarism in academia

The problem of plagiarism according to Ryan (2007) is widespread in the academic world; moreover it is not tied to specific variable like background, ethnic group, culture, gender or any other attribute. Plagiarism in classrooms has changed a lot, the practices that were used on period of 70's are very different from the current ones; in the past word-for-word copy in papers was mainly from one or two sources which made it easy for instructors to discover although it is time-consuming. Nowadays with the advent of the world-wide-web the attitude of the plagiarism among students and instructors has changed as students started to use this tool as a shortcut to their assignments, instructors started to use it to police student activities.

Influences affecting such behavior are mainly caused by the actors involved in the process; who are the faculty and the student; Broeckelman (2008) found that instructors who are aware of plagiarism and who implement tools and methods for detection and who discuss this problem with their students and who emphasize the source attribution and proper citation are those who are likely to discover such behavior among students. Also, students are less likely to report their engagement of serious plagiarism if the instructors spend time discussing plagiarism with students. Moreover, students are affected hardly by other students behavior in a way that more students are expected to commit plagiarism if other students are practicing any activity of academic dishonesty.

On the other hand, Jian et al.(2008) found that instructor impact is smaller than expected and that the course difficulty is the main reason that makes students plagiarize as a policy followed by different kinds of students. Vogts (2009) suggests that students use plagiarism as a last resort especially in

programming assignments that are difficult to students and urge that students behavior is not the same when students are novice programmers hence plagiarism detection tools shouldn't be used just to detect offenders and punish them but to discover when novice programmers need more help and thus more education support.

2.2. Plagiarism Detection Tools

Many sophisticated plagiarism detection tools and algorithms are designed and used nowadays to limit the problem; most of them were a result of an academic research rather than designed solely for commercial purposes. Some tools have proposed algorithms that can handle all kinds of digital submissions from plain text, formatted text to audio podcast by transforming all forms of submissions to plain text, and then apply the famous Winnowing algorithm to search for text similarity with other texts which can be done locally or globally.(Butakov & Scherbinin, 2009). Other natural language based techniques (White.& Joy., 2004) were implemented in the direction of peer-to-peer sentence comparison using techniques that can help solving obfuscation resulted from paraphrasing , reordering, merging or splitting sentences. Each pair of sentences is given a similarity score and a common score using natural language algorithms to measure similarity and then each pair is checked and stored in Links DB, see Figure-1.

Pseudocode for the Comparison Algorithm

```
Document[] docs = readDocsFromDisk();

for each Document, i, in docs {
  for each document,j, following i in docs{
    compareSentences(docs[i], docs[j]); }}

compareSentences(Document doc1, Document doc2){
  for each sentence,i, in doc1{
    for each sentence,j, in doc2 {
      int common = number of shared words;
      int score = similarityScore(i,j,common);
      if(score > SIM THRESHOLD ||
        common > COM THRESHOLD)
        storeLink(sent1, sent2, score); }}}}
```

Figure-1: Pseudo code for the Comparison Algorithm (White.& Joy., 2004)

Using sentence based comparison can increase the affectivity of uncovering plagiarized text.

Most tools can be categorized into web-enabled tools and stand alone tools, Turnitin and EVE2 are examples of the most popular tools used of both types, next a detailed discussion about each:

1. Web enabled tools

Web enabled tools take the benefit of offering the ability to be accessed by any machine, anytime anywhere the service is needed, *Turnitin* as evaluated by Alzahrani et al., (2009), is the most well-known commercial plagiarism detection tool to which many universities from UK and USA subscribe. It uses an enormous database from the Internet and previous student works to be compared with the query document. Researchers (Butakov. & Scherbinin., 2009 ; Kaner., 2008) found that Turnitin works by comparing writings to articles in its

proprietary database and in some commercial or academic databases. It is mainly used on the domain of plain text using the technique of fingerprint (Ryu. et al., 2008). Turnitin is known for its large databases of student papers (over 40 million documents as of January 2008). The Turnitin service can be used as an add-on for many Course Management Services (CMSes), but two requirements present a barrier to its adoption. First, the institution must pay for the proper Turnitin subscription, which can be a problem for small educational organizations. Second, there must be a stable 24/7 network connection between the CMS and the Turnitin servers. However Turnitin is not free and takes long time to get results (Lukashenko. et al., 2007).

How Turnitin works?

Documents submitted to Turnitin are compared with billions of papers published on the internet, local database or registered

academic databases of journals and periodicals (Figure-2).

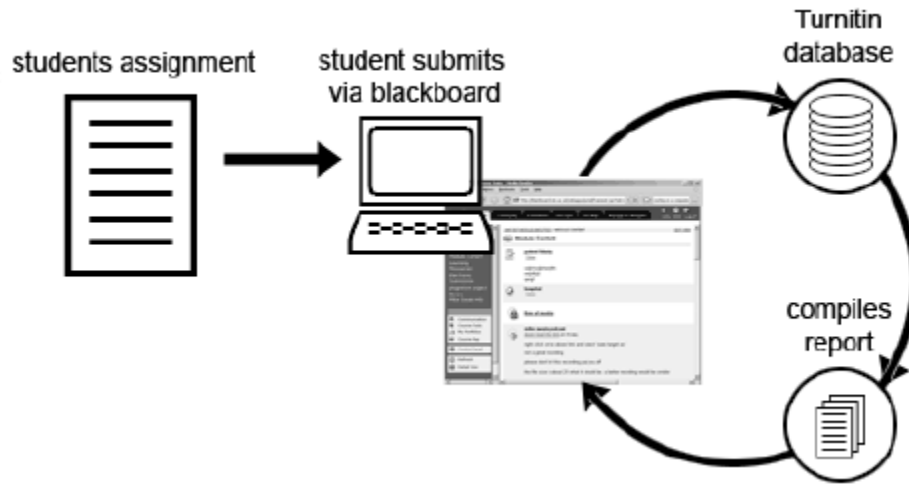


Figure-2: How Turnitin works (Jones., 2008)

Each submitted document is checked for originality by means of text matching comparison with a huge list of documents, and then report results in the “originality

report”(Figure-3) that specify the percentage of copy ignoring the common words like articles.

author	title	report	gm	file	paper ID	date
Anonymous	341721.doc	100%	--	341721.doc	1061546	03-07-07
Anonymous	341740.doc	100%	--	341740.doc	1061554	03-07-07
Anonymous	341720.doc	100%	--	341720.doc	1061630	03-07-07
Anonymous	346450.doc	99%	--	346450.doc	1061600	03-07-07
Anonymous	343041.doc	97%	--	343041.doc	1061568	03-07-07
Anonymous	346590.doc	96%	--	346590.doc	1061587	03-07-07
Anonymous	346594.doc	96%	--	346594.doc	1061605	03-07-07
Anonymous	343026.doc	96%	--	343026.doc	1061606	03-07-07
Anonymous	346438.doc	92%	--	346438.doc	1061615	03-07-07
Anonymous	342620.doc	90%	--	342620.doc	1061592	03-07-07

Figure-3: Originality Report (Jones., 2008)

2. Stand-alone tools

Desktop application that are installed and run in individual PCs, EVE2 (Essay Verification Engine) for example, has the capability to make large number of searches on the Internet to locate matches between sentences in the query document and potential resources websites (Alzahrani. et al., 2009). Thus, in order for EVE2 to work, the machine should be connected to the internet; it is used in the domain of plain text

using the technique of web search (Ryu. et al., 2008). As Turnitin it works in the domain of plain text and is not free, however it has no internal database; it depends on the internet to get instant results and it is designed for the use of teachers only, i.e. not for students (Roxas., 2006).

Although these sophisticated tools are quite popular, where Turnitin.com has over 4000 clients in the US Colleges alone, other tools like EduTie and EVE2 are quite popular too but Howard. (2007) argues that still the free

searching using search engines like Google seems the best and most accurate choice for many reasons:

- Its free
- Its accuracy is comparable to available tools
- It avoids problems caused by other tools of copyright and violation of student intellectual property rights when such tools requires contribute student work into its service.

2.3. Source Code Plagiarism

Source code plagiarism is extensively researched in literature, since it is an intellectual property and a human artifact that needs to be protected against theft and because the plagiarism is more conducted in this area.

Many algorithms are proposed to solve the problem of source code plagiarism; one of them introduced by Lim et al. (2009) which works by identifying programs through establishing “birthmark” for each one; which means specifying the characteristics of the program that make it identifiable by its own and use this birthmark as a means of comparison among other programs.

This algorithm considers the control flow information of the program which shows not only the structural characteristics of the program but also the behavior of this program.

Asymmetric distance measure is another algorithm by Ji. et al (2007) that computes the distance of authorship between different programs through estimating the evolution of a set of similar program clones by establishing a phylogenetic tree representing common structural and behavioral features shared by different program clones.

Some tools can be categorized under Structure-metric systems that extract and compare representations of the program structures. Therefore, they give an improved measure of similarity and have more effective and practical techniques to detect program plagiarism, such as MOSS, YAP, SIM, Pdetect and SID (Zhang. et al., 2007).

MOSS system uses Winnowing algorithm to select and compare document fingerprints. It is insensitive to white space, has noise suppression and has position independence properties. The main steps of Winnowing algorithm are:

(1) divides programs into k-grams, which is a continuous substring of a user-defined length k (2) computes the hash of each k-gram, (3) selects a subset of these hashes to be the program’s fingerprint, and (4) compares the fingerprints. YAP3 system, the third version of YAP, converts programs into token strings and uses Greedy-String-Tiling (GST) algorithm to compare them. GST algorithm’s aim is to find a maximum set of common continuous sub-strings as long as possible, and then count the number of single-line differences within blocks of matching tokens. SIM system also tokenizes the programs, and compares token sequences by means of dynamic programming string alignment technique. PDetect system supposes that plagiarism program causes no major changes in the set of keywords (Zhang. et al., 2007).

3. Methodology

The Project (Anti-P) is a desktop application that has no internal database, it opens documents of the types (*.txt, *. doc, *.pdf and *.htm) subject to check for originality and extracts the following from the document Text:

- Title
- Authors

- Keywords

Search queries will be passed to Google search engine with smart choices by the program itself to minimize the search time and get quick results thus a clear proof of plagiarism if any. The main algorithm that our project will use rely on natural language processing techniques to be able extract the main parts of the document, then pass arbitrary parts of the document as search queries to Google and receive results in hash tables, the decision of plagiarism conduct will be based on the results stored on hash table.

The behavior of the program is illustrated in Figure-4.

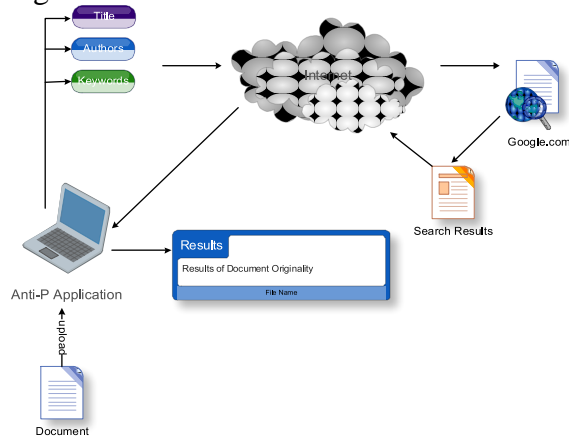


Figure-4: How Anti-P works

The Program activities can be illustrated in the following diagram:

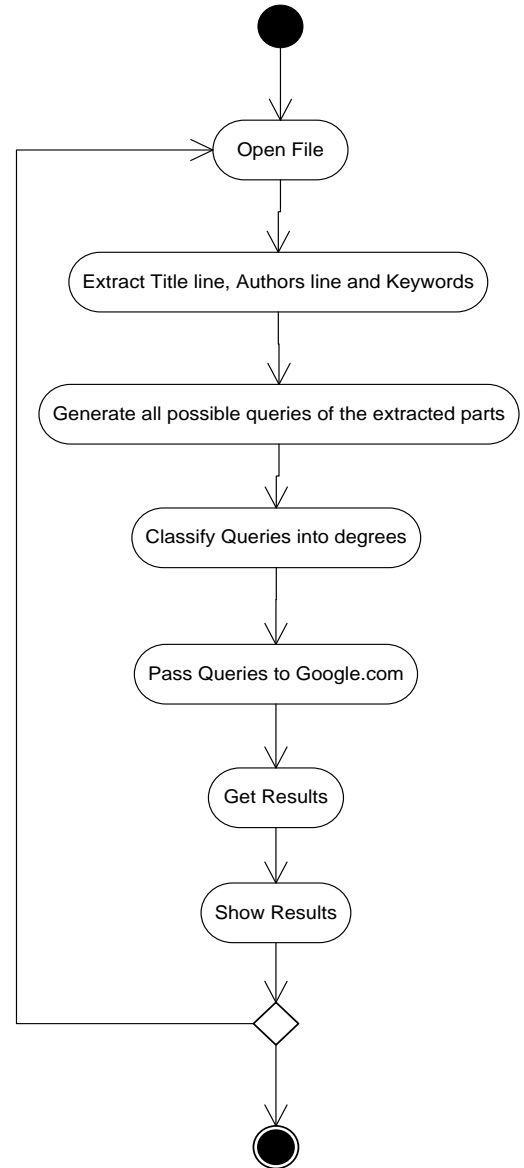


Figure-5: General flow of activities performed by Anti-P to detect plagiarism

Query Generation and Categorization is done by connecting the three parts using the connectors “AND” and “OR” with different combinations which will affect the degree of how strict match we are imposing on the search engine to get results. Generating all possible combinations requires designing an algorithm that combines all parts of query at run time using a 2-dimentional binary matrix -of dimension $[2^{(n-1)}] [(n-1)]$ where n is

the number of keywords- that represent the connectors of the query parts (operators), to better understand the flow of work suppose we are to generate queries for document that has 3 keywords, the program generates the a binary matrix [4][2]:

$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$$

Algorithm 1: Generate Binary Matrix

```
Rows = 2 ^ (Number of Keywords -1 )
Cols = Number of Keywords -1
Declare matrix[Rows][Cols]
For(i=0 to Rows-1){
matrix[i]= BinaryRepresentation[i]}
```

Figure-6: Generate Binary Matrix Algorithm

Each 0 represents “AND” and each 1 represents “OR”, Figure-7 illustrates how queries are generated.

Algorithm 2:Generate Query Combination

```
Rows = 2 ^ (Number of Keywords -1 )
Cols = Number of Keywords -1
String Query [Rows];
For (i=1 to Rows){
Query[i]= QueryParts[0];
For(j=1 to Cols){
If(matrix[i][j]=='0')
Query[i]="AND"+QueryParts[j+1]
Else
Query[i]="OR"+QueryParts[j+1]} }
```

Figure-7: Generate Query Combination Algorithm

Then queries are categorized according to the number of ANDs and ORs that actually represent how strict the query is:

Algorithm 3: Categorize Queries

```
For(i=1 to Rows){
If( Query[i].ConnectorTitleAuthor =
"AND")
Category Color= Red
Else
If(Query[i].Count(AND)>=
Query[i].Count(OR) )
Category Color=Orange
Else
Category Color=Yellow }
```

Figure-8: Categorize Queries Algorithm

Table-2 shows the queries for a document that has 3 keywords and their categories. T for Title, A for Author and Kn for Keyword words number n, if the document has 3 keywords for example, the queries are:









Query	Degree
("T" and "A") and ("K1" and "K2" and "K3")	 High Possibility
("T" and "A") and ("K1" and "K2" or "K3")	 High Possibility
("T" and "A") and ("K1" or "K2" and "K3")	 High Possibility
("T" and "A") and ("K1" or "K2" or "K3")	 High Possibility
("T" or "A") and ("K1" and "K2" and "K3")	 Moderate Possibility
("T" or "A") and ("K1" and "K2" or "K3")	 Moderate Possibility
("T" or "A") and ("K1" or "K2" and "K3")	 Moderate Possibility
("T" or "A") and ("K1" or "K2" or "K3")	 Low Possibility

Table-2: Queries and their classifications for a three key worded research paper

The Document text will be colored according to plagiarism degree, also results of queries and the links to suspected sources will be shown.

4. Experiments And Evaluation

Evaluating this tool will be through preparing a data set (files submitted by students) that are to be checked for originality and use our application to test the degree of Plagiarism the application has classify each file.

4.1.Preparing the Data for Evaluation

The files used in the test are categorized into five categories as demonstrated in Table-3:

Category No.	Category	Number of files
1	Title and Author and Keywords are plagiarized	4
2	Title and Author only are plagiarized	2
3	Title and Keywords are plagiarized	2
4	Author and Keywords are plagiarized	2
5	Title only plagiarized	3
6	Author only plagiarized	4
7	Keywords plagiarized	2
8	None of the three is plagiarized	5

Table-3: Files and their categories that were used for experiment

Categorizing files that way tests not the categorization part but also the extraction of those three parts. Each category is given a number that represents the degree of plagiarism to be able to represent it using charts. See Table-4.



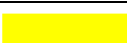
Degree	Value
	10
	8
	6
None	4

Table-4: Plagiarism degrees and their corresponding values

4.2.Evaluation Results

After executing the program and passing all the files, the degree specified for each file is computed with the result of the rest of the files in the same category hence each category is assigned a number that represent an average of the degrees returned.

Category No.	Average Degree Specified
1	9
2	8
3	8
4	7
5	7
6	5
7	3
8	0

Table-5: Experiment results by files categories

Using our application with the mentioned data set revealed the following:

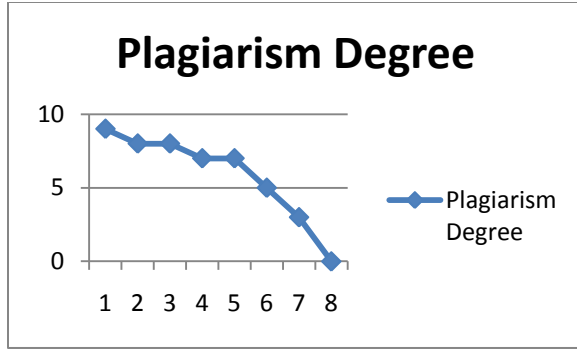


Figure-9: Chart demonstrating the relation between Plagiarism degree specified by Anti-P and files categories

Note: All files used in this test have the Title in the first line, Author names in the second line.

As the results shown in Figure-9, many comments can be drawn:

- Plagiarism Degree curve is downward sloping which is expected since categories are numbered in descending order.
- The first category where all parts are plagiarized didn't hit the red category in most of the files which means a weakness of passing very strict queries and in criteria that didn't meet the red category.
- The last category that all parts are not plagiarized was not categorized as a plagiarized document even that these files contains plagiarized body which was not discovered by our program.
- On average and working under the assumption that the parts used to detect plagiarism are Title, Author and Keywords, results are very satisfying and expected.

Recommendations that can be concluded according to experiment results are:

- ✓ Increase the number of queries will give wider range for queries categorization thus enhance accuracy of the plagiarism degree.

- ✓ Use the full text search will exceed the thoroughness and the accuracy of the search process by far, since every word of the document is checked against plagiarism.
- ✓ Enhance extraction techniques will make the application robust and reliable to detect plagiarism of any document regardless the structure of text inside it.

Comparing the project to Turnitin, the project is thought to give more accurate results, using more resources, and cost less space and processing time.

	Turnitin	Anti-P
Need Database	Yes	No
Sources	Limited	Unlimited
Response time	Relatively long	Relatively small

Table-6: Comparison between Turnitin and Anti-P

The Project deals with unstructured data, it directly extracts search terms from the opened file, unlike Turnitin where documents are stored on special databases and every term in the paper is classified, also as discussed earlier Turnitin use its own database of documents and registered to special academic databases which means that if some document plagiarize from a document that is not from Turnitin resources, it will not be caught by Turnitin as a plagiarized document, while Anti-P has the ability to reach all resources that is available to Google.com.

In terms of processing time, Turnitin (Butakov & Scherbinin, 2009) requires long time processing documents, while Anti-P requires the time needed by Google to get results.

5. Conclusion and Future

5.1.Conclusion

We have described how plagiarism can be detected through the automation of passing text to search engines and get results, which opens the door to a free, precise and efficient plagiarism detection methodology, although the project is in its initial phases, experiment results shows how promising this approach can be if it is extended to consider full text in its search process. This is especially important for institutions with limited funds to afford expensive plagiarism tools and that have huge number of students.

As mentioned many times earlier the Project in its current situation uses only Title, Authors and Keywords for detection plagiarism which makes our project ideal for journals and organizations that accepts academic papers and needs to check if they are already published before or if the same author published the paper in different title or same title with different author, it is not mature yet for universities use because it is not using full text in the search process.

Our initial evaluation results showed that most categories of plagiarism can be detected easily by the tool but for some of them the tool did not detect the accurate degree of plagiarism and that can be justified because the tool did not dig deep into the full text of the documents.

5.2. Future Work

Current Project depends on the three elements of any document to detect plagiarism-that are-Title, Author and Keywords, future work will expand the source of the queries to include the whole text of the document line by line, and then to recursively take different combinations of words in the same line after removing articles, pronouns and connecting words that thought to be irrelevant. Also we are planning to improve the techniques used in the process of extraction as one of the most difficult challenges we are facing is to cope

with all different formats of research articles and papers by using advanced and specialized algorithms and tools.

Another major future goal is to build a library of research documents similar to a crawler that is dedicated to this purpose to allow some advanced options such as incremental queries, cache results, and a rich decision support system.

5.3. Research Issues

- Crawling deep web documents and overcoming some websites restrictions, files securities, etc.
- The frequent and automated use of Google.com violates its Terms of service which may block the search page for a while.
- The application performance is hardly affected by the internet connection speed.
- No wide and enough supporting tools to extract text properties like font size and others from different file formats.

6. References

- [1] S. Butakov and V. Scherbinin, "The toolbox for local and global plagiarism detection", *Computers & Education* 52 (2009) 781–788.
- [2] S. Alzahrani, N. Salim and M. Alsofyani, "Work in Progress: Developing Arabic Plagiarism Detection Tool for E-Learning Systems", 2009 International Association of Computer Science and Information Technology - Spring Conference, 978-0-7695-3653-8/09 (2009) IEEE
- [3] D. Vogts, "Plagiarising of Source Code by Novice Programmers a "Cry for Help"?", SAICSIT'09, 12-14 October 2009, Riverside, Vanderbijlpark, South Africa. Copyright 2009 ACM

- [4] H. Lim, H. Park, S. Choi and T. Han, "A method for detecting the theft of Java programs through analysis of the control flow information", *Information and Software Technology* 51 (2009) 1338–1350.
- [5] K. Jones, "PRACTICAL ISSUES FOR ACADEMICS USING THE TURNITIN PLAGIARISM DETECTION SOFTWARE", *International Conference on Computer Systems and Technologies - CompSysTech'08*.
- [6] M. Broeckelman, "Faculty and Student Classroom Influences on Academic Dishonesty", *IEEE TRANSACTIONS ON EDUCATION*, VOL. 51, NO. 2, MAY 2008
- [7] H. Jian, F. Sandnes, Y. Huang, L. Cai and K. Law, "On Students' Strategy-Preferences for Managing Difficult Course Work", *IEEE TRANSACTIONS ON EDUCATION*, VOL. 51, NO. 2, MAY 2008
- [9] C. Ryu, H. Kim, S. Ji, G. Woo and H. Cho, "Detecting and Tracing Plagiarized Documents by Reconstruction Plagiarism-Evolution Tree", 978-1-4244-2358-3/08 (2008) IEEE
- [10] J. Ji, G. Woo, and H. Cho, "A Plagiarism Detection Technique for Java Program Using Bytecode Analysis", *Third 2008 International Conference on Convergence and Hybrid Information Technology*, 978-0-7695-3407-7/08 (2008) IEEE
- [11] C. Kaner, "A Cautionary Note on Checking Software Engineering Papers for Plagiarism", *IEEE TRANSACTIONS ON EDUCATION*, VOL. 51, NO. 2, MAY 2008, 0018-9359/ (2008) IEEE
- [12] R. Lukashenko, V. Gaudina and J. Grundspenkis, "Computer-Based Plagiarism Detection Methods and Tools: An Overview", *International Conference on Computer Systems and Technologies - CompSysTech'07* (2007) ACM ISBN: 978-954-9641-50-9.
- [13] J. Ryan, "Plagiarism, Education, and Information Security", PUBLISHED BY THE IEEE COMPUTER SOCIETY 1540-7993/07 (2007) IEEE.
- [14] J. Ji, S. Park, G. Woo and H. Cho, "Understanding the Evolution Process of Program Source for Investigating Software Authorship and Plagiarism", 2007 IEEE.
- [15] L. Zhang, Y. Zhuang and Z. Yuan, "A Program Plagiarism Detection Model Based on Information Distance and Clustering", 2007 *International Conference on Intelligent Pervasive Computing*. 2007 IEEE
- [16] R. Howard, "Understanding "Internet plagiarism"", *Computers and Composition* 24 (2007) 3–15, 2007 Elsevier Inc.
- [17] R. E. Roxas, N. R. Lim and N. Bautista, "Automatic Generation of Plagiarism Detection Among Student Programs", 1-4244-0406-1/06 (2006) IEEE
- [18] D. WHITE, M. JOY, "Sentence-Based Natural Language Plagiarism Detection", *ACM Journal on Educational Resources in Computing*, Vol. 4, No. 4, December 2004. Article 2.
- [19] A. Rovai, "Online and traditional assessments: what is the difference?", *Internet and higher Education* 3 (2000) 141-151, 2001 Elsevier Science Inc.