

Plagiarism Detection Using Natural Language Processing Techniques

N. Malik¹, A. Bilal², M. Ilyas³, S. Razzaq⁴, F. Maqbool⁵, Q. Abbas⁶

¹Department of Computer Science, Virtual University of Pakistan, Jinnah Campus, Lahore, Pakistan.

²School of Electrical Engineering and Computer Science, NUST, Islamabad, Pakistan.

^{3,4,5,6}Department of Computer Science and Information Technology, University of Sargodha, Sargodha, Pakistan.

³Muhammad.Ilyas@uos.edu.pk

Abstract- Nowadays, plagiarism became very common in many fields of life, such as research and educational fields. Due to the advancement in plagiarism techniques adopted by plagiarists, it is challenging to detect plagiarism accurately by the existing techniques. Different features are observed while checking plagiarism, such as syntactic, lexical, semantic, and structural features. This research explores new and modern plagiarism detection tasks especially, text-based plagiarism detection including monolingual plagiarism detection. We proposed a four-stage novel framework for plagiarism detection. This framework uses natural Language Processing (NLP) instead of focusing on traditional string-matching approaches. This framework explores text similarity by combining two metrics as Skip-Gram and Dice Coefficient, on the corpus-based approach. The Deep meaning of the text is explored by the use of the Deep and shallow NLP technique. Our results conclude that Heavy revision is identifying quickly through Deep NLP. Shallow NLP prepares text very well that is processed further quickly. Word2vec results are close to simple Deep NLP methods, but word2vec also highlights those documents that other techniques may not highlight. Synonym and phrase changes are also captured through Deep NLP.

Keywords- Skip-Gram; Natural Language Processing; Word2vec; Word Embedding; Deep NLP; Shallow NLP.

I. INTRODUCTION

Plagiarism is the demonstration of using another person's unique words and thoughts as one's ideas and viewed as an ethical offense [2]. Unoriginality definition in the sense of lexicon is "The move or routine regarding making another person's work or thought, and showing as one's own; artistic burglary " (Oxford English Dictionary 1).

Such type of action becomes a very intention-seeking problem since the foundation of training appraisal. Since we entered the web period, it becomes easy for getting a lot of data quickly, and chances of copy-pasting or stealing and using others' text become easier [1],[3],[6]. It is an illegal deed used to make others work as their property without any proper references. Plagiarism makes individuals mindless, lazy, and thief in their academic or practical life [9]. Plagiarism is related to those problems that are very complex and complicated and very demolish in electronic media. As internet usage increases day by day, plagiarism is also increased. In the case of paraphrasing, sentence restructuring and missing references make this problem more complex [3][10].

A few procedures, counting tokenization and parameterized coordinating, were created to battle with such changes [2][4]. In any case, comparative techniques are harder to apply for regular writings, so it isn't easy to use this technique to overcome the unoriginality of text. The term reporting is also used in plagiarism using different alternative words or synonyms to avoid the exploration of plagiarism by the software detectors. Many people do not know that they use plagiarism act such as our religious scholars uses others text but not reference it. Merriam-Webster first time defines plagiarism word, and after that, laws are built to stop such activity [7].

Plagiarists use word-level and phrase-level plagiarism, and changing the sentence structure becomes more challenging to explore. Tools used to detect plagiarized work have been developed and used for this purpose, free and paid. However, such tools still have many weaknesses to detect phrase changes or sentence rewriting cases accurately. The unavailability of an accurate framework behind current tools encourages us to work on this issue. Our research work tries to find a solution that overcomes all existing challenges and detects plagiarism in monolingual cases.

We present a novel framework that uses Natural Language Processing (NLP) technique to explore

complex plagiarism more accurately, involving skip-gram, Dice metric, and word embedding Deep NLP technique. This framework includes four stages; preprocessing, similarity comparison, filtering using Deep NLP, and document classification using machine learning. Our research work explains that Word2vec results are close to simple Deep NLP methods, but word2vec also highlights those documents that other techniques may not highlight.

The remainder of the paper contains the following sections; Section 2 contains the literature review. Section 3 focuses on problems and their solution. Section 4 describes the Similarity metric. Section 5 describes the preprocessing of Shallow and Deep NLP. Section 6 illustrates classification, results, and discussion, and Section 7 concludes the overall work.

II. RELATED WORK

So far, many different techniques are developing to identify plagiarism. Many researchers put their methods and techniques to solve plagiarism detection problems. To detect plagiarized documents, many attempts have been made earlier. Most plagiarism detection techniques utilize string processing algorithms, i.e., these methods are used to find the identical string's occurrence within the document. Some work is also done in NLP plagiarism detection techniques. Some work in plagiarism detection using other than the proposed framework is discussed below in this section.

Vani and Gupta discussed different plagiarism detection techniques considering improved ranking based semantic, syntactic based, semantic-based, fuzzy-based, metrics-based, etc. Their experiments show that SRL (Semantic Role Labeling) gives average results, including both recall-precision. Adding a sentence ranking feature in SRL-R (Semantic Role Labeling with Sentence Ranking) accuracy of SRL has been more valuable [13]. The SRL-R has better accuracy and precision while testing on fewer amounts documents, so there is a need to test it on an extensive collection.

SVD (Singular value decomposition) also gives reasonable outcomes, but these results belong to those tests carried on Arabic text documents [13]. FUZZ (Fuzzy Based Semantic String Similarity) doesn't give improved results in both cases. Contrarily, In the case of PDLK (Plagiarism Detection using Linguistic Knowledge) gives the best outcomes of precision, although intermediate results are obtained in terms of recall [13]. These techniques required some additional functionality to reduce computation overhead and enhance accuracy and precision. The author also did not test them on a large corpus, so the outcomes produced at a small corpus may be dramatically different in the case of a large corpus [13].

Strilețchi introduced a cross-platform framework used to detect similarities among different source codes. The author makes a detection environment that permits the quantified correlations that work for enhancing the cognizance among developers about the importance of originality at the time of the code development process [8]. This tool finds the analog among source files specially written in Java, C/C++, JavaScript, and PHP languages. Authors' work also develops awareness about originality and their writing along with similarity detection. The technical University of Cluj-Napoca is selected to test and develop the above tool to detect the originality of programming students' assignments [8]. By introducing the above tool in checking students' assignments, the instructor can find original and plagiarized work and mark assignments with accurate marks [8].

Sharma and Jindal used a custom search engine that implements the crawling process and presents a technique involving extra-corporal semantic matching. Author access global data to perform semantic analysis through the source link. Their method generates very efficient and accurate results.

Halak and El-Hajjar introduced two techniques related to prevention and plagiarism detections. In the first approach, a unique assignment is allocated to each student to focus on his work without copy and stilling others' assignments. So in this situation author get a reduction in plagiarism cases, while in another case, each student allocates the individual presentation of coursework collections [9]. In group-based projects, the individual presentation gives effective results in the plagiarism detection process. In this type of assignment, students are shuffled and divided into groups. They are allocated different assignments where each group solves the task in a software or hardware system [9].

Krizkova et al. used multiple classifications of detection engines to show the range and practical nature of PD systems. The author chose eight articles from various plagiarism detection fields, and after an in-depth analysis, he compares them among detection systems and various writing formats [10]. The author uses these eight selected unmodified papers for detection engines' detection base at the first stage. Then he uses a modified version of the above-selected papers by modification of plagiarized documents and sentences that are explored in the first stage [10].

Ming proposed a deflection setting program similarity exploration approach that is best for whole program PD. The author uses the same library code rewritten and different programs to test the presented approach's working and efficiency [11]. Partial and total program plagiarism is efficiently detected through LoPD.

Bagai proposed a new text-based plagiarism approach. Their approach is more efficient than classical approaches because their approach first considers

cluster formation, including those documents that are more similar than using a hybrid algorithm based on substring [12]. After the application of this algorithm, they use keyword similarity.

Vani and Gupta's study focused on extrinsic PD of various related or join similarity metrics and shows the effectiveness of combined similarity metrics instead of single congenital metrics [13]. They also analyzed the usage and influence of speech tagging (POS) in PD systems. They introduce the union of various four (Match coefficient Dice coefficient, Fuzzy-Semantic measure and, Cosine similarity) metrics [13].

Wibawa proposed a system that is based on MapReduce and Hadoop Framework to compute the similarity index of documents through the Ferret algorithm is more suitable than others. Author studies show that a system that works with Hadoop System is slower than those run on standalone computers [14]. While the case of those systems having 200MB or more documents on Hadoop runs more speedily as compared to stand alone. Standalone systems are made to be run in either OS, while the Hadoop system can work only with OS [14].

Muhammad et al. presented a technique of PD that compares the potential documents with sources searched from MEDLINE. Their method also searches those plagiarized documents produced by replacing words or phrases in the sentences from original texts [15].

Ning examined four different keyword elution methods for Weighted TF, IDF, TF-IDF, and TF-IDF relies on the section. They chose VSM (Vector Space Model), an IR-based model for the elution of keywords [16]. Their study proves that both the same keyword data through different methods and the same methods for dissimilar data types produce remarkably different outcomes [16]. Hurtik and Hodakova focused on the detection of plagiarized images from the corpus. F-1 transform method is used to explore plagiarized images through FTIP [17]. This is used at the preprocessing step, and this step is very. This step is performed only at the beginning at one time in the whole process to limit the transaction. Infact it is the step that only increases the overall speed of the image searching process [17].

Sindhu and Idicula presented a detection system working on Malayalam text-based collection by using the fingerprinting PD technique. Malayalam contains a highly complex language structure as it is a rich source of morphological and compound words and proves to be a challenging task for PD [18]. The author introduces a method for PD of Malayalam that builds limitations of similarity among various documents. In this method, the author uses a winnowing algorithm at the sentence level for measuring fingerprints [18].

Hiremath and Otari research covered three different approaches used for plagiarism detection [20]. A text-based PD system is used to detect simple plagiarism types such as simple paraphrased text and passages. But

these methods are not able to explore long paraphrased and translated plagiarism [20]. The second method is citation-based, which is used to explore citation base plagiarism [20]. Zou et al. presented a clustering PD system used at SCUT's learning management system and used to explore plagiarism in engineering students' assignments [21]. This method has three steps; pre-selecting, locating, and post-processing. In Pre-selecting, by using the same fingerprints, select potential documents at the limited collection [21]. Alzahrani and Salim worked on PD algorithm that works with fuzzy semantic-based string similarity technique and works at four stages [4]. Pre-processing is the first stage and incorporates the removal of stemming, tokenization, and stop words. In the second stage, the Jaccard coefficient and shingling are used to list candidate documents against each potential document [4]. Potential documents are now analyzed in a sentence manner with source documents. This comparison is made with a fuzzy degree of similarity [4]. In the last stage that is also called post-processing, neighbor sentences are combined to make one sentence. The author's results show that 54% of detection is correct, but only 30% of plagiarism cases are tested [4].

Adam's work showed that after making changes in NLP's existing PD algorithm, it works very accurately [22]. The addition of Semantic Parsing enhances the ability of plagiarism detection. In the detection of similar text, the addition of Syntactic Parsing works POS-tagger [22]. The author also discusses the rearrangement of words at the start of similarity detection decreases the overall time [22]. The proposed framework also uses POS tagging.

Gupta research work describes the external plagiarism detection technique based on NLP and Fuzzy semantic approach. The author's primary focus is on exploring POS tagging, stop word removal, and lemmatization [23]. POS tagging is used to compute the n-gram similarity measure. Their work also provides a base for the utilization of Fuzzy semantic with NLP techniques. They use a Fuzzy semantic measure that provides more accurate results and improved performance [23].

Chong proposed a framework that uses Natural Language Processing techniques. The author's research focuses on a corpus-based approach to reflect the usage and need of text preprocessing, deep, and shallow methods. The author divides the technique into two steps to assess the working of the technique [24].

Korpai and Bose formed a framework by combining supervised machine learning and NLP for plagiarism detection [25]. They form this framework for monolingual plagiarism detection. They introduce an n-gram frequency comparison approach for efficient detection [25]. One hundred twenty characteristics are involved in this framework used at the time of the preprocessing step through the NLP technique. The next

step is filtration used to filter similar words, and then the document is arranged at four levels according to plagiarism level using a supervised classification learning algorithm [25]. Then false negative and positive are computed using the Confusion matrix. The author's framework gives an impressive result with 89% false negative and positive ratio [25].

Chong shed light on plagiarism detection using natural language processing techniques. The author discusses the effectiveness of NLP in plagiarism detection [26]. He also explains the usage of High-level natural language processing techniques that are beneficial for improving current techniques of PD [26]. The results show that besides other techniques, NLP provides more accurate results and performance [26].

Zouaq presented the NLP technique for ontology learning. The author uses shallow and Deep NLP techniques [27]. The author's work includes lexico-syntactic patterns that are study through Deep NLP analysis. The author uses knowledge reasoning, computational linguistics, and knowledge representation to make computational semantics [27]. Their corpus selection is web base.

Ceska and Fox work describes that the preprocessing step in PD does not work well with detection accuracy [28]. Author work shows that the factor that improves the performance of PD are synonym recognition (SYR), number replacement (NMR) while the word generalization (WG) little bit enhance the performance [28]. Author experiments also show that instead of a longer N-gram, stop word minimizes the processing time while lemmatization works to improve run time [28].

Clough presented a report on different natural and programming language methods for plagiarism detection [29]. The author highlights all techniques for PD in NLP briefly. His report shed light on current techniques that are very effective in plagiarism detection [29]. He also discusses these techniques with the help of different examples [29].

Mozgovoy's work described the NLP parser in plagiarism detection. The author says that using the NLP parser is very good in swap word detection but not very good in similar documents [30]. Swap between two files that use the same citation is very low as compared to other techniques. The presence of swap words in any text can work as the primary key to detect plagiarism in that text [30]. The author further adds that short string sentences are effective in finding swap words, but false matches are also increased in this case [30].

Ezzikouri's work include the introduction of a programming interface application for multilingual words and concepts that contain different Semantic Similarity metrics measuring semantic similarity/distance [31]. The author uses the WordNet dictionary for translation from monolingual to

multilingual (English-French and English-Arabic) cases [31]. The author also introduces a GUI written in for accessing the covenantal system [31].

Hattab uses LSI (Latent Semantic Indexing) for the production of encounter-language semantic space. English and Arabic papers are used to explore how it checks the context alikeness [32]. Outcomes of the author's experiment prove that this method works very effectively at a 93% similarity level in detecting plagiarism in the English-Arabic language [32].

Pinto proposed the M1 model used to detect multilingual plagiarism [33]. They show how this model work to tackle the task such as cross-lingual analysis, categorization, Information retrieval, etc. [33]. Their results prove the effectiveness of their proposed model. Ceska proposed a technique MLPlag for multilingual plagiarism detection [35]. The proposed method works in plagiarism detection by matching the position of the word. The author uses EuroWordNet thesaurus to translate Multilanguage to mono language form [35]. Detection of potential text derived from a source document by translating into a different language becomes easy [35]. The author uses the semantic-based word normalization method.

Lee developed a method that explores text-similarity in multilingual plagiarism. They develop a platform that detects plagiarism in the multilingual collection [36]. They collect Chinese-English collections from the internet and instruct text classifiers through the Vector Machine model [36]. After instructing, classifies arranged non-familiar text according to them and gets arranged text using Self-Organizing Maps [36]. Using this methodology, they evaluate similarity accurately.

III. PROBLEMS AND THEIR SOLUTION

A. Natural Language Processing technique

The main goal of this research is to examine the NLP methods in content reprocess recognition. The idea is that rewritten and original text have some similar pattern explored by the depth analysis [6],[26]. To check the similarity pattern, a new system is proposed having NLP procedures including shallow NLP and Deep NLP with more advanced methods such as word2vec. Each rewritten and original source writings are in English monolingual composed writings form. The corpus-based approach is used to assess the system by viewing different documents through different angles.

In education, plagiarism spoils the skills, academic writing, and critical analysis of students and deprives the students of hardworking and thinking abilities [9]. Reducing false exploration and improve efficiency in the considerable corpus is very is a significant challenge. The understanding complex part of NLP covers a highly complex area of AI (artificial intelligence). NLP provides its application in almost every part of life, such

as advertisement, email handling, customer care, web search, language transcription, translation, etc. [6]. A variety of machine learning models and NLP-related tasks are implemented in almost various programming languages. Shallow and Deep learning techniques in NLP give better results as compared to other techniques [51]. These models cannot depend upon classical task-specific methodology and work as a single end-to-end model.

B. A proposed framework for plagiarism detection

An earlier study by Chong shows that the introduction of NLP (Natural Language Processing) provides more accurate outcomes when applied to paraphrased texts. NLP work has no experiments basis, but NLP in plagiarism detection (PD) is inspired by earlier work carried in this field and is suitable for many collections [24]. Candidate separation and preprocessing are the main parts of all PD systems. In preprocessing, we can generalize the text while candidate separation minimizes the overall time for exploration to speed up the analysis stages [24]. At different steps of plagiarism detection above technique is applied. The Deep NLP technique is implemented at the deeper analysis stage of PD. In contrast, some text preprocessing stage uses a shallow NLP technique that is very simple with the very least resource utilization, i.e., stop word elimination/removal, tokenization, stemming, lemmatization, and lower casing [35]. The proposed framework is divided into four distinct stages. Fig. 1 shows the overall working of the proposed framework. The proposed system is derived from the classical three-stage approach of PD. The system's work is focused on the type of input data, although not every stage needs task-specific assignments. Stages of the proposed framework are;

- 1st stage: Pre-processing
Both source text and potential suspicious documents are prepared for subsequent stages. Text preprocessing and shallow NLP (Natural Language Processing) techniques are applied to the texts.
- 2nd Stage: Similarity comparison using skip-gram and Dice metric
A pairwise comparison among source and suspicious documents is carried out using skip-gram and Dice coefficient metric. These computed scores are then passing to the next stage for further processing.
- 3rd Stage: Filtering through Deep NLP
First of all Deep NLP technique, along with the word2vec process, is evaluated, and after that, candidate pairs are generated. The analysis of similarity scores generates the candidate pair by introducing a threshold point at each similarity score.
A pair that matched with the selected threshold are marked as candidate pair [15]. In this way, those

documents having complex plagiarism ignored due to low similarity values are also explored by evaluating Deep NLP first. Then with the help of the threshold point, candidate documents are selected.

- 4th Stage: Classification

After selecting candidates and similarity scores from the above stage, the selection of plagiarized or none plagiarized text is marked here. We can use the WEKA tool for the classification of documents.

IV. SIMILARITY METRIC

The similarity measure is a significant and essential task in many techniques such as document categorization, information retrieval, and plagiarism detection [36]. We can select the desired document by using similarity measurement. There are enormous similar measurement methods and tools that are currently ruled by computer science. Similarity measurement is actively involved in other fields like speech recognition, machine translation, IR, bioinformatics, dialectology, and lexicography [19][37].

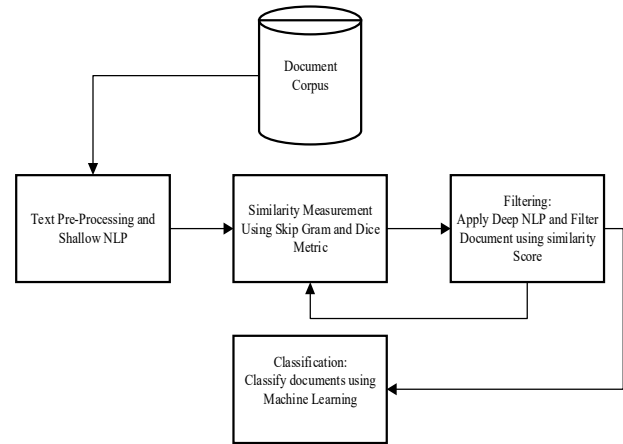


Figure 1: Proposed framework for plagiarism detection

Almost every plagiarism detection technique uses n-gram for similarity exploration. This metric is straightforward and easy to use, but a big issue is the data sparsity problem [36][39]. This metric also requires components of string to be consecutive that reduces the token of pairs. To overcome data sparsity and the consecutive problem, we use Skip-gram that is the generalization of the n-gram. Skip-gram is the generalized form of the n-gram. It is not consecutive like n-gram and skips spaces over a string by leaving gaps [38]. K-skip-n-grams is define as to be the set $w_{i1}, w_{i2}, \dots, w_{in}$

$$\sum_{j=1}^n ij^{-ij^1} < k \quad (1)$$

Much experimental evidence proves the Dice similarity coefficient metric's efficiency after testing Jaccard, Overlap, Cosine similarity, and Dice similarity coefficient measures. We compare similarity values of Cosine, Dice, Jaccard Overlap, and Dice coefficient metric. By analyzing similarity values, we come to know that the Dice metric produces better results compared to other metrics on the small corpus. By keeping the benefits of both Dice and skip-gram metrics, we decided to propose a modified version of the dice similarity coefficient that will work with skip-gram to produce more accurate results than other techniques. Table I shows the similarity values of a small corpus having ten documents.

TABLE I: SIMILARITY VALUES OF DIFFERENT METRICS

Data Set	Overlap Coefficient	Cosine Similarity	Jaccard Index	Dice Coefficient
g0pA taska	39.93%	36.20%	36.18%	36.84%
g0pA taskb	39.95%	35.60%	35.02%	35.14%
g0pA taskc	36.03%	79.87%	79.20%	80.87%
g0pA taskd	33.68%	90.97%	90.30%	91.97%
g0pA taske	32.45%	98.01%	97.33%	98.71%
g0pB taska	31.63%	85.54%	86.45%	86.54%
g0pB taskb	27.96%	74.12%	73.70%	75.12%
g0pB taskc	34.21%	71.36%	70.36%	72.36%
g0pB taskd	23.86%	60.70%	60.60%	61.70%
g0pB taske	27.88%	71.18%	71.10%	72.16%

We process our text in skip-gram as following:

Input text: Such people have to suffer insults and indignities patiently. It becomes difficult for him to give up."

Potential suspicious text: The man in high office has no freedom or time to imagine himself. He has no strength over himself. Such people have to undergo affront and indignities patiently. It becomes challenging for him to surrender."

Skip-gram for source text: 1 skip 2 gram will be:[the in], [man high], [in office], [high has], [office no], [has liberty], [no or], [liberty time], [or to], [time think], [to of], [think himself], [he no], [has power], [no over], [power himself], [Such have], [people to], [have suffer], [to insults], [suffer and], [insults indignities], [and patiently], [it difficult], [becomes for], [difficult him], [for to], [him give], [to up]

Skip-gram for suspicious text: 1 skip 2 gram will be; [the in], [man high], [in office], [high has], [office no], [has freedom], [no or], [freedom time], [or to], [time imagine], [to of], [imagine himself], [he no], [has strength], [no over], [strength himself], [such have], [people to], [have undergo], [to affront], [undergo and], [affront indignities], [and patiently], [it hard], [becomes for], [hard him], [for to], [him surrender]

After producing a skip-gram, there is a need for a similarity metric that counts similarity measures

between these tokens. Due to accurate similarity measurement compared to other metrics, the Dice similarity coefficient is used in this research to produce similarity outcomes in source and suspicious documents. After preprocessing and shallow NLP, that prepare the text for further processing; Dice similarity coefficient with skip-gram is used to compute lexical generalization, constituent extraction, Predicate generalization, and predicate extraction are evaluated Dependency relation extraction. We use a small corpus of 95 documents build by Clough and Stevenson [50] and produce desire similarity scores using the above metric. The output generated at this stage provides the foundation for categorizing four levels of plagiarism, which has already been discussed [25]. Table II shows the similarity values of the corpus. Corpus contains five source documents. Five groups of students were asked to answer that is also included in five source documents.

TABLE II: SIMILARITY VALUES

Group A	Group B	Group C	Group D	Group E
0.3684	0.8554	0.9929	0.8909	0.7328
0.3514	0.7512	0.987	0.9675	0.7981
0.8087	0.7236	0.5163	0.4134	0.7278
0.9197	0.617	0.9382	0.5991	0.6409
0.9871	0.7216	0.5643	0.7706	0.6476
0.3812	0.5974	0.8644	0.726	0.7213
0.3201	0.7115	0.9537	0.6069	0.8129
0.917	0.6346	0.6702	0.5444	0.8768
0.3075	0.8971	0.9932	0.9884	0.8247
0.2396	0.7366	0.838	0.8012	0.8961
0.8775	0.8319	0.8228	0.7241	0.8114
0.9509	0.7024	0.3084	0.9997	0.7809
0.3448	0.8948	0.9827	0.6668	0.6158
0.6079	0.7642	0.8966	0.9175	0.6976
0.6045	0.6852	0.7136	0.9517	0.6967
0.3246	0.657	0.9564	0.9755	0.9715
0.9924	0.7427	0.8059	0.6742	0.8524
0.8671	0.5614	0.7995	0.5917	0.8733
0.2532	0.8922	0.7227	0.6012	0.5174

V. PREPROCESSING, SHALLOW AND DEEP NLP

A. Preprocessing and Shallow NLP

Natural Language Processing Toolkit12 (NLTK), CoreNLP, and OpenNLP provide the preprocessing stage techniques. In this stage, text development and analysis are carried out carefully. These techniques include [40]:

- Sentence segmentation,
- Tokenization,
- Stop word removal,
- Punctuations removal,
- Lower casing and
- Number replacement

Shallow NLP purely deals with the structure of sentences, but it is not suitable for analyzing the syntactic and semantic characteristics of the text. We use this technique from the OpenNLP toolkit, NLTK toolkit, or the Stanford CoreNLP toolkit. Mostly shallow NLP technique includes [35][41]:

- Part-of-speech tagging (POS),
- Lemmatization,
- Stemming,
- and chunking

B. Deep NLP

Deep NLP is applied by exploring the deep knowledge like a quantifier extent, anaphora resolution [27]. Complete sentence meanings are explored by computational semantics. The syntactic parser is the fundamental component of deep NLP that works with syntactic grammars [27]. Deep NLP includes:

- Syntactic constituent exploration,
- Dependency relation exploration,
- Lexical generalization,
- Predicate extraction,
- Named entity extraction,
- Predicate generalization.

Word embedding is also studied in Deep NLP and provides very accurate results as compared to other techniques.

We use Open NLP and Core NLP toolkit to evaluate syntactic constituent exploration, Dependency relation exploration, Lexical generalization, Predicate extraction, Named entity extraction, and Predicate generalization.

C. Word Embedding

NLP provides a mapping of words into vectors; representation of words allows the facility to work more efficiently with machines as these provide numeric values that are easily understood by machines [42]. Word embedding is an essential concept of NLP contains similar illustration with related meanings of words [42]. This is related to feature learning and language modeling, including mapping words and phrases in sentences to related vectors, belongs to the actual numbers family [42]. Mapping involves a mathematical embedding process.

Mapping is carried out through probabilistic models, dimensionality reduction, co-occurrence matrix, and neural networks [43]. It is used to increase the working of NLP, most likely when we do sentiment analysis and syntactic parsing [43]. In the corpus of data, word embedding is used to explore semantic similarities among documents by focusing on linguistic characteristics.

Word2vec is a word embedding toolkit developed by Tomas Mikolov [44]. Primarily new word embedding techniques work by using neural that more efficient than n-gram models [44]. In the word2vec model, our focus is on a single word, and we try to guess all those words that are possibly present nearby the surrounding of this word [45]. It works as a representative layer in the architecture of Deep learning and convert isolated word into a positional representation concerning other words in corpus and used well in different NLP methods [45]. It provides a similarity of text in terms of syntactic and semantic dimensions [46]. We store information related to our desire words but do not alter dimension capacity it would permeably be 25 to 1000 dimensions.

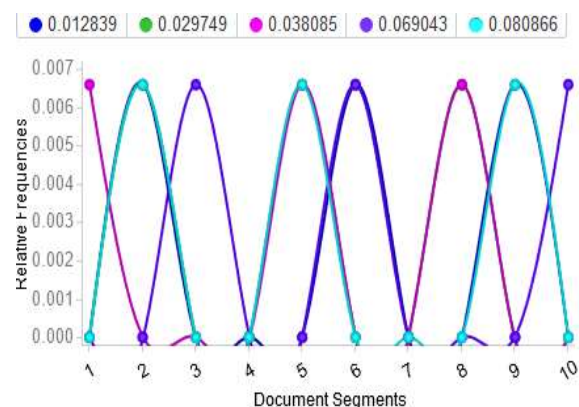


Figure 2: Phrases Correlation

It also uses deep neural architectures that are best for contextual entity linking, constituency parsing, writing style recognition, sentiment analysis, etc. [46]. Target words are explored through Neural Network, whose hidden layer encodes the word representation [47]. Word2vec is implemented through two models that are Skip-gram and Continuous Bag of Words (CBOW). Here Skip-gram model is used [45].

The Skip-gram model is more efficient because it tackles multiple vectors, producing more word vectors [46]. We precede our work by taking the sentence, "Such people have to suffer insults and indignities patiently." Now, if we focus on the first three words, then our window's size would be three and denoted by "m."

Such people have to suffer insults and indignities patiently

Figure 3: Testing Phrase

An additional neural network layer is also present equal in dimension with embedding dimension but is also smaller than the vector size of input and outcomes.

Finally, in the output layer, a function called softmax activation function is used, so we find how likely a word appears surrounding our desired word. Neural networks follow the fake task to predict the nearest words. We get word embedding by fetching hidden layers. Fig. 4 shows similar word vectors produced through the word2vec method.

Next, we select our word that is the main focus and predict its surroundings here, and we select a word to suffer that is the center word of the sentence. We try to predict all words that come after and before the word "suffer".

inheritance 1	
estate	0.5700975132989379
inherit	0.4872831946145624
dowry	0.43965497780921287
wealthy	0.4293107568019739
divorce	0.427726370285732
wealth	0.4217992165182588
tax	0.4196920602069459
husbands	0.41670886858159983
heir	0.403674277817153

Figure 4: Similar Word Vectors

We do this by using the formula given below [48];

$$J(\theta) = \frac{1}{T} \sum_{i=1}^T \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{t+j} | w_t) \quad (2)$$

Another formula is also used to compute probabilities that are given below [46][44];

$$\log p(o | c) = \log \frac{\exp(u_o^T v_c)}{\sum_{w=1}^W \exp(u_w^T v_c)} \quad (3)$$

Fig. 2 shows phrases correlation, and Fig. 5 shows document text correlation computed through the word2vec method.

D. Comparative Analysis of the Proposed Framework

The basic framework includes three main stages preprocessing, filtering, and detection. First of all, the document is prepared for processing, then similarity computed, at the last documents are classified as plagiarized and non-plagiarized [26].

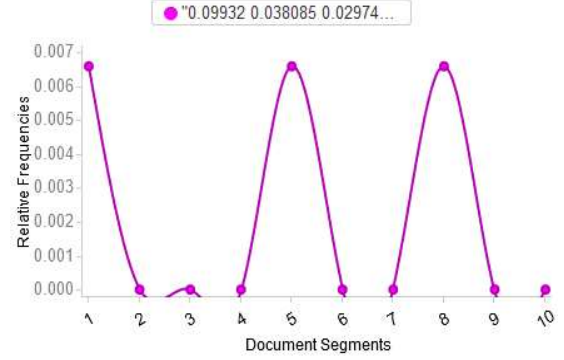


Figure 5: Document Text Correlation

There is another framework proposed by [26]. This framework includes five stages. That includes (1) Pre-processing, (2) Similarity Comparison, (3) Filtering, (4) Further processing, and (5) classification.

The Deep NLP technique is applied at stage four in the existing framework after the filtering stage (where candidate pairs are generated). After filtering, Deep NLP is evaluated, so there may be some suspicious documents that are ignored due to low similarity values but also contains complex plagiarism.

At stage two, similarity comparisons are taken through n-gram and Jaccard coefficient. But with n-gram, we get consecutive tokens, so the number of tokens reduced.

Some document is ignored due to low similarity values and contains complex plagiarism (synonym and paraphrasing) and does not take part in the Deep NLP technique.

Based on similarity scores, only those documents with more similar values are extracted, and then Deep NLP features are applied, as shown in Table III. In this case, documents having complex plagiarism but does not have apparent sufficient similarity values are ignored. Word embedding technique is not used in this framework.

On the other hand proposed framework includes four stages. These stages are:

- (1) Preprocessing
- (2) Similarity Comparison
- (3) Filtering
- (4) Classification

Pre-processing and classification stage is the same, but stage two, where similarity is computed, is different. Instead of using traditional n-gram, skip-gram with Dice coefficient is used that produced more token pairs. After all, it does not generate consecutive tokens, so the chance to get more similarity values increases.

In the proposed system at the filtering stage, the Deep NLP technique along with the word2vec process is evaluated, and after that candidate, pairs are generated. In the proposed system, we separate Deep NLP features then-candidate pairs are generated, so the chances for complex plagiarism detection (like synonym and

paraphrase changes) become more. And false detection may also be reduced. By using the word2vec method, the deeper meaning of the text is explored, and all hidden plagiarism cases are highlighted.

VI. CLASSIFICATION, RESULTS, AND DISCUSSION

The last step of our experiment is to use a machine learning classifier to classify each document. WEKA tool is used to classify suspicious documents. It supports classification, clustering, and preprocessing tasks and includes various learning algorithms such as linear machine learning and nonlinear machine learning algorithms [49]. We get top documents having more similarity with source as candidate documents at the filtering stage. These candidates' documents are fed into WEKA for classification. Gain Ratio attribute evaluator is used to extract more occurring features.

TABLE III: DEEP NLP FEATURES CORRELATION

Number of Features	Technique	Correlation %
152	word2vec synonym and paraphrase	0.82
63	Syntactic constituent extraction	0.77
49	Lexical generalization	0.79
4	Predicate extraction	0.78
5	Named entity recognition	0.81
48	Dependency relation extraction	0.77

We get results that show that features obtain from shallow NLP and ordinary Deep NLP technique having more similarity values are very close to word2vec features values. Deeplearning4j tool on WEKA is used to implement the word2vec method on the corpus. Words are converted into vectors, and then the Deep learning tool analyzes text for synonym and phrase similarity between the source and suspicious documents. The WEKA toolkit provides Deeplearning4j. This is a deep learning programming library and is implemented by Java language.

It provides many Deep learning algorithms such as deep belief net, deep auto-encoder, recursive neural tensor network, restricted Boltzmann machine, GloVe, stacked denoising auto-encoder, word2vec, and doc2vec. We use the word2vec machine learning algorithm and find the semantic and paraphrasing changes among source and suspicious documents. The deep neural classifier is trained in this purpose, and document semantic and paraphrase similarity values are evaluated in terms of recall, correlation, accuracy, etc.

Fig. 6 shows a simple example of vectors produced from corpus documents. Word2vec also explores those documents that contain complex plagiarism and should not be highlighted by an ordinary method. When the Deep NLP technique is evaluated, the next step is to apply filtering on documents and fetch out desired candidate documents. These are used for similarity computation. We select those documents as candidate documents that are more plagiarized at the plagiarism filtering stage. The similarity is computed by comparing source document text with potential suspicious document text. After applying the similarity metric, we get top documents having more similarity with the source as candidate documents. These documents were going to the next phase, where they are feed-in machine learning classifiers. Gain Ratio attribute evaluator is used to extract more occurring features.

TABLE IV: CLASSIFICATION ACCURACY

Feature Class	Accuracy %
Word2vec features	74
Deep NLP feature set	72.7
All features set	68.1

Naive Bayes updateable classifier algorithm classifies documents according to their precision, accuracy recall, and correlation values. This algorithm is the extended version of the naïve Bayes algorithm. It uses estimator classes to classify attributes. Table IV displays classification accuracy, recall, and precision.

0.26818 0.14346 -1.3777 -0.10866 -0.23201 0.012839 -0.27877 0.016257 0.11384 0.69923 -0.51332 -0.47368 -0.33075 -0.13834 0.2702 0.30938 -0.45012 -0.4127 -0.09932 0.038085 0.029749 0.10076 -0.25058 -0.51818 0.34558 0.44922 0.48791 -0.080866 -0.10121 0.38097 -0.46508 3.8463 0.31362 0.13643 -0.52244 0.3302 0.33707 -0.35601 0.32431 0.12041 0.3512 -0.069043 0.36885 -0.67368
0.25168 -0.24517 0.25381 0.1367 -0.31178 -0.6321 -0.25028 0.25818 0.17346 -0.278768 0.016567 0.14384 0.69583 -0.61332 -0.83075 -0.93834 0.2202 0.30938 -0.43012 -0.3027 0.012839 -0.09932 0.038085 0.029749 0.10076 -0.25058 -0.51818 0.34558 0.44922 0.48791 -0.080866 -0.10121 -1.3777 -0.10866 -0.23201 -0.46508 3.8463 0.31362 0.13643 -0.52244 0.3302 0.33707 -0.35601 0.32431 0.12041 0.3512 -0.069043 0.36885 0.012839 -0.32901
0.78168 -0.20517 0.87381 0.1367 -0.87178 -0.6921 -0.25028 0.79097 0.67818 0.20346 -0.72877 0.016257 0.90384 0.70923 -0.67332 -0.87368 -0.89075 -0.85834 0.6562 0.87568 -0.95015 -0.6721 -0.09932 0.038085 0.029749 0.10076 -0.25058 -0.51818 0.34558 0.44922 0.48791 -0.080866 -0.10121 -1.3777 -0.10866 -0.23201 -0.46508 3.8463 0.31362 0.13643 -0.52244 0.3302 0.33707 -0.35601 0.32431 0.12041 0.3512 -0.069043 0.36885
0.76185 -0.65473 0.86754 0.76543 -0.87632 -0.7351 -0.65426 0.98765 -0.98765

Figure 6: Vectors of words extracted from a corpus

Naïve Base updateable classifier is similar to Naïve Bayes classifier, but it has Kernel density estimation methods instead of normal density measures. We use Gain Ratio Attribute Evaluator to extract features pair

having more part in similarity values combination. This attribute is used to assign a ranking to the documents according to their similarity score [1]. It computes the part of an attribute in the document by measuring its gain ratio according to its class. Table V shares feature generated by the Gain Ratio Attribute evaluator.

TABLE V: FEATURES GENERATED BY GAIN RATIO ATTRIBUTE EVALUATOR

Number of Features	technique	Correlation %
14	Lemmatization with dice skip-gram	0.64
48	Dependency Relation	0.77
63	Syntactic Constituent extraction	0.77
4	Predicate extraction	0.78
5	Named entity recognition	0.81

TABLE VI: F-SCORE, RECALL, AND PRECISION OF DOCUMENT PLAGIARISM CATEGORY [DA-DEEP ANALYSIS, WF-WORD2VEC FEATURE]

Plagiarism Category	F-Score %		Precision %		Recall %	
	DA	WF	DA	WF	DA	WF
Light Revision	44.5	45.8	43.6	54.3	96.9	96.8
Near Copy	58.5	60	66.9	67	52.9	53
Heavy Revision	49.3	61	50.5	53.9	46.8	72
Non-plagiarized	90.5	94.8	52.5	90.9	96.9	97

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have shown the basic working of our proposed framework. Based on an experiment, we conclude that Heavy revision is identified easily through Deep NLP. Shallow NLP prepares text very well that is processed further quickly. The result generated from the machine learning tool helps us identify and classify all four categories of plagiarism cases. Almost all documents are classified easily, but only 2 to 3 documents are not classified correctly. Word2vec results are close to simple Deep NLP methods, but word2vec also highlights those documents that other techniques may not highlight. Synonym and phrase changes are also

captured through Deep NLP. Results are shown in Table VI.

We use a small corpus that produces results very close to actual cases, but it might be a different case other type and size of corpus evaluated. Because we have not more resources to do our experiments on a larger scale, we limit our results at the current stage. Our results provide a base for further investigation for researchers. These are initial results; to prove them there would be a lot of work is needed.


Numbers of the experiment are required to check the validity of the concern approach. We want to extend our approach on a significant corpus level in the future to produce a more effective framework that enhances the capability of existing tools such as Turnitin etc. In this research, we only identify plagiarism and classify it according to its contribution to similarity values.

REFERENCES

- [1] Z. Tian, Q. Wang, C. Gao, L. Chen and D. Wu. Plagiarism Detection of Multi-Threaded Programs via Siamese Neural Networks, in IEEE Access, vol. 8, pp. 160802-160814, 2020.
- [2] D. Santos and D. Ferreira. Plagiarism detection based on blinded logical test automation results and detection of textual similarity between source codes, IEEE Frontiers in Education Conference (FIE), Uppsala, 2020, pp. 1-9.
- [3] Q. Huang, X. Song and G. Fang. Code Plagiarism Detection Method Based on Code Similarity and Student Behavior Characteristics, ICAICA, China, 2020, pp. 167-172.
- [4] S. Alzahrani and N. Salim. Fuzzy Semantic-Based String Similarity for Extrinsic Plagiarism Detection, Braschler, and Harman, Vol. 1176, 2010, pp. 1-8.
- [5] N. Khan, C. Agrawal and H. Yadav. An Effective Compressive Sensing based N-gram Approach for plagiarism detection, 2nd IDEA, Bhopal, India, 2020, pp. 1-7
- [6] T. K. Arachchi and E. Y. A. Charles. Deep Learning Approach to Detect Plagiarism in Sinhala Text, 14th ICIIS, Kandy, Sri Lanka, 2019, pp. 314-319
- [7] Z. Li, Z. Zhu and T. Yang. A Multi-index Examination Cheating Detection Method Based on Neural Network. 2019 IEEE 31st ICTAI, Portland, OR, USA, 2019, pp. 575-581
- [8] C. Strilețchi, M. Vaida, L. Chiorean and S. Popa. A cross-platform solution for software plagiarism detection, 2016 12th IEEE ISETC, Timisoara, 2016, pp. 141-144
- [9] B. Halak and M. El-Hajjar. Plagiarism detection and prevention techniques in engineering

- education, 2016 11th EWME, Southampton, 2016, pp. 1-3
- [10] S. Krizkova, H. Komaskova and M.Gavalac. Preference comparison for plagiarism detection systems. In Proc. of IEEE International Conference on Fuzzy Systems, 2016.
- [11] J. Ming, F. Zhang, D. Wu, P. Liu and S. Zhu. Deviation-Based Obfuscation-Resilient Program Equivalence Checking With Application to Software Plagiarism Detection, in IEEE Transactions on Reliability, vol. 65, no. 4, 2016.
- [12] M. Bagai, Vibhanshu, S. Gupta and R. Ali. Text based plagiarism detection. International Journal For Technological Research In Engineering Volume 3, Issue 8, 2016.
- [13] K. Vani and D. Gupta. Investigating the impact of combined similarity metrics and POS tagging in extrinsic text plagiarism detection system, 2015 ICACCI, Kochi, 2015
- [14] C. Wibawa, I. Bastian and M. Mustikasari. Document Similarity Measurement Using Ferret Algorithm and Map Reduce Programming Model. IJCTT, 19(2):76-82, 2015
- [15] R. Muhammad, A. Nawab, M. Stevenson and P.Clough. An IR-based Approach Utilizing Query Expansion for Plagiarism Detection in MEDLINE. ISBN: 1545-5963
- [16] H. Ning, L. Kong, M. Wang, C. Du and H. Qi. Comparisons of Keyphrase Extraction Methods in Source Retrieval of Plagiarism Detection. In Proc. of 4th ICCSNT, 2015. China.
- [17] P. Hurtik and P. Hodakova. FTIP: a Tool for Image Plagiarism Detection. In Proc. of SOCPAR. 2015.
- [18] L. Sindhu and S. M. Idicula. Fingerprinting based detection system for identifying plagiarism in Malayalam text documents, 2015 CoCoNet, Trivandrum, 2015, pp. 553-558
- [19] M. Ilyas and J. Kung. A Similarity Measurement Framework for Requirements Engineering, 4th ICCGI, Cannes, 2009
- [20] S. Hiremath and M.S. Otari. Plagiarism Detection-Different Methods and Their Analysis: Review. IJIRAE, 2014.
- [21] D. Zou, W. Long and Z. Ling. A Cluster-Based Plagiarism Detection Method. GuangZhou, China. Lab Report for PAN at CLEF, 2010.
- [22] A. Adam and Suharjito. Plagiarism detection algorithm using natural language processing based on grammar analyzing, Indonesia, 2014
- [23] D. Gupta, K. Vani and C. K. Singh. Using Natural Language Processing techniques and fuzzy-semantic similarity for automatic external plagiarism detection, 2014 ICACCI, New Delhi, 2014, pp. 2694-2699
- [24] M.Y.M. Chong. A Study on Plagiarism Detection and Plagiarism Direction Identification Using Natural Language Processing Techniques. 2013. Semantic Scholar, Corpus ID: 61166267
- [25] R. Korpai and S. Bose. A Framework for Detecting External Plagiarism from Monolingual Documents: Use of Shallow NLP and N-gram Frequency Comparison. Pune, India, 2010.
- [26] M. Chong, L. Specia and R. Mitkov. Using Natural Language Processing for Automatic Detection of Plagiarism. University of Wolverhampton, UK. 2010
- [27] A. Zouaq. Shallow and Deep Natural Language Processing for Ontology learning: a quick overview. Canada, 2010.
- [28] Z. Ceska and C. Fox. The Influence of Text Pre-processing on Plagiarism Detection. In Proc. of International Conference RANLP 2009 – Borovets, Bulgaria, pages 55–59.
- [29] P. Clough. Plagiarism in natural and programming languages: an overview of current tools and technologies. University of Sheffield, UK. 2000.
- [30] M. Mozgovoy, T. Kakkonen and E. Sutinen. Natural Language Parsers in Plagiarism Detection. Finland, 2007
- [31] H. Ezzikouri, M. Oukessou, and M. Erritali. Semantic Similarity/Relatedness for Cross-language plagiarism detection. In Proc. of 13th International Conference Computer Graphics, Imaging, and Visualization. Morocco, DOI 10.1109/CGIV.2016.78.
- [32] E. Hattab. Cross-Language Plagiarism Detection Method: Arabic vs. English. In Proc. of International Conference on Developments in E-Systems Engineering. Dubai, UAE. 2010. DOI:10.1109.
- [33] D. Pinto, J. Civera, A. B. Cedneo, A. Juan and P. Rosso. A Statistical Approach to Cross lingual Natural language task. Mexico. 2009. DOI:10.1016/j.jalgor.2009.02.005.
- [34] M. Waseem, Q. Abbas, M. Ilyas, S. Razzaq, S. Niazi and I. Asghar. Classifying Urdu Verbs Using Rule Based Approach. LGURJCSIT 21, Vol. 5, No. 1, pp. 68-75, 2021.
- [35] Z. Ceska, M. Toman, and K. Jezek. Multilingual Plagiarism Detection. AIMSA, pp. 83–92, 2008.
- [36] A. Akinwale and A. Niewiadomski. Efficient Similarity Measures for Texts Matching. Nigeria. Vol. 23 No. 1, 2015, pp. 7-28.
- [37] G. Kondrak. N-Gram Similarity and Distance. Edmonton, Canada. LNCS 3772, pp. 115–126, 2005.
- [38] D. Guthrie, B. Allison, W. Liu, L. Guthrie and Y. Wilks. A Closer Look at Skip-gram Modeling. LREC, 2006.

- [39] M. Ilyas and J. Kung. A comparative analysis of similarity measurement techniques through SimReq framework. FIT 2009, ACM, Vol. 9, issue 47, 2009, Pakistan
- [40] M. Cheatham and P. Hitzler. String Similarity Metrics for Ontology Alignment. 2013. USA. DOI https://doi.org/10.1007/978-3-642-41338-4_19.
- [41] M. Chong and L. Specia. Lexical Generalisation for Word-level Matching in Plagiarism Detection. In Proc. of PAN-10. 2011.
- [42] C. Dima and E. Hinrichs. Automatic Noun Compound Interpretation using Deep Neural Networks and Word Embeddings. Germany. In Proc. of 11th International Conference on Computational Semantics. 2015.
- [43] J. Wieting, M. Bansal, K. Gimpel and K. Livescu. CHARAGRAM: Embedding Words and Sentences via Character n-grams. USA. 2016. arXiv:1607.02789v1.
- [44] T. Mikolov, I. Sutskever, K. Chen and G. Corrado. Distributed Representations of Words and Phrases and their Compositionality. 2013. arXiv:1310.4546[cs.VL].
- [45] X. Rong. Word2vec Parameter Learning Explained. 2016. arXiv:1411.2738.
- [46] S. Jansen. Word and Phrase Translation with word2vec. 2018. CoRR abs/1705.03127.
- [47] P. Suhartono, A. Iskandar, M. I. Fanany and R. Manurung. Utilizing Word Vector Representation for Classifying Argument Components in Persuasive Essays. 2016. Indonesia.
- [48] J. Cross, B. Xiang and B. Zhou. Good, Better, Best: Choosing Word Embedding Context. 2015. arXiv:1411.2738v4.
- [49] S. R. Kalmegh. Comparative Analysis of WEKA Data Mining Algorithm Random Forest, Random Tree and LAD Tree for Classification of Indigenous News Data. 2015. India. ISSN 2250-2459.
- [50] P. Clough and M. Stevenson. Developing A Corpus of Plagiarised Short Answers. Sheffield S1 4DP, UK. Language Resources and Evaluation: Special Issue on Plagiarism and Authorship Analysis, 2010. [Download]. DOI 10.1007/s10579-009-9112-1.
- [51] J. Haneczok and J. Piskorski. Shallow and deep learning for event relatedness classification. Information Processing & Management Volume 57, Issue 6, 2020.

Paper Titled: Plagiarism Detection Using Natural Language Processing Techniques			
Certificate			
<p>The subject article has not been published or submitted to or accepted for publication in any form, in any other journal or conference etc. Subject article is our own original contribution. We also guarantee that the authorship of this article will <i>not</i> be contested by anyone whose name(s) is/are not listed by us here and we will not request the authorities of journal to add any more author(s) after submission.</p> <p style="text-align: right;"><u>Muhammad Ilyas</u> Corresponding Author</p>			
Authorship and Contribution Declaration			
Sr.#	Author-s Full Name	Contribution to Paper	Signature
1	Nasreen Malik (Main/principal Author)	Proposed topic, basic study Design, methodology and manuscript writing	
2	Ahmad Bilal. (2 nd Author)	Study design and concept, algorithm formulation	
3	Muhammad Ilyas (3 rd Author)	Literature review & Referencing, Drafting	
4	Saad Razzaq (4 th Author)	literature search, data cleaning	
5	Fahad Maqbool (5 th Author)	Data analysis and interpretation, quality insurer	
6	Qaisar Abbas (6 th Author)	data collection , Referencing, and quality insurer	