JOURNAL OF MECHANICS OF CONTINUA AND MATHEMATICAL SCIENCES
www.journalimcms.org

ISSN (Online) : 2454 -7190 Vol.-17, No.-12, December (2022) pp 14-30 ISSN (Print) 0973-8975

GRAVITY SCORE: NEW METRIC TO MEASURE PLAGIARISM IN TEXT DOCUMENTS USING THE CONCEPT OF GRAVITATIONAL FORCE

Srijit Panja

Data Scientist, Cognida.ai, Hyderabad, India.

Email: srijit.panja@cognida.ai

https://doi.org/10.26782/jmcms.2022.12.00002

(Received: September 12, 2022; Accepted: December 5, 2022)

Abstract

Present-day computational capabilities allow digital assets like images, videos, text, and audio have features comparable to those in real-world entities. Location is one such aspect. Similar to real-world bodies being represented by vectors on cartesian coordinates, digital media entities (like text, as discussed in this paper) when encoded, each component of the encoding representing a feature, conceptually should have a vector representation in each such encoding. The concept is put to practice by text encodings (embedding) techniques like Bag-of-words, TF-IDF, Word2Vec, Glove, and Transformer models like BERT, AlBERT etc which create vectors out of the text. This paper aims to use a combination of features in text analogous to mass and distance and propose a new plagiarism index cloning the formula of gravitational force. Parameters like the length of documents/number of words, semantics, frequency of each word, etc, one or many of which are often missed out in prevalent algorithms of text similarity calculations, are important for detecting and measuring plagiarism. The paper aims to consider all such possible parameters in the formulation of a new plagiarism metric to be coined as Gravity Score.

Keywords: Natural Language Processing, Text Embedding, Text token, Gravitation

I. Introduction

Between two bodies that have masses greater than zero, the force of attraction keeps getting higher as they keep getting closer to each other. This is according to Newton's law of universal gravitation. Correspondingly two closely spaced text encodings or almost similar sentences or phrases can be perceived to be attracted strongly towards one another. Therefore if we were to consider a hypothetical force existent in the system, it would be very strong. This value or as we propose 'Gravity score' should be an output from a formula that is analogous to the formula of Gravitational force, and therefore should consider parameters comparable to masses of real-world bodies as well. In the text domain, these could be replaced by stakeholders *(Chujo and Utiyama 2005)* like 'number of words', 'number of characters', 'number of sentences', 'number of characters except stopwords and white spaces' etc. Text similarity calculation using cosine similarity, depending on how the texts are encoded

considers syntax and semantics to judge how similar two texts are. Conversely, Jaccard Similarity measures sameness based on frequency of each constituent character. The idea to introduce this new metric different from state-of-the-art text similarity calculations like Cosine similarity, Jaccard similarity, Compression similarity, Pairwise similarity, etc is aligned with the aim to consider all lexical, syntactic, and semantic parameters involved. Accurate plagiarism detection and calibration are subject to all such aspects.

II. Concept of Gravitation and derivation of Gravity Score for Text Similarity

According to Newton's law of universal gravitation *(Verlinde2011)*, every particle in the universe attracts every other particle with a force that is directly proportional to the product of their masses and inversely proportional to the square of the distance between their centers *(Fock 2015)*. This is expressed by the formula:

$$F = G\frac{m_1 m_2}{r^2} \tag{1}$$

where F is the force between the particles, G is the gravitational constant = $6.674\times10{-}11 m^3 k\ g^{-1} s^{-2}$, m_1 and m_2 are masses of two bodies and r is the distance between their centers.

These masses of bodies are comparable to the bulk of the text *(Nation and Waring 1997)*, which can be quantified in several ways. Let us denote these values as b_1, b_2 for two documents. These documents can be encoded using various word or text embedding techniques that convert words to word vectors. The Euclidean distance *(Danielsson1980)* between these data points corresponding to the vectors if represented as d , the above formula modifies to

$$F = \frac{b_1 b_2}{d^2} \tag{2}$$

In this case, we obliterate G as F doesn't denote Gravitational force here, but is a proposed text similarity metric to quantify plagiarism. Consequently, we can replace F with S to denote the similarity value that we call Gravity Score. Besides, prime modification in the numerator would be to replace the bulk of the text with relative bulk *(Cattaneo1958)* to confine value from the numerator within a range. Consequently, to procure a Gravity Score within a scale of 0 to 1 depending on the extent of copying from one document to another, a bias of 0.25 is added to the denominator. This modifies the formula to:

$$S = \frac{\frac{b_1}{b_1+b_2}\times\frac{b_2}{b_1+b_2}}{d^2+c} \tag{3}$$

where $0 \le b_1/(b_1+b_2)$, $b_2/(b_1+b_2) \le 0.5$. $b_1/(b_1+b_2)$ and $b_2/(b_1+b_2)$ are the relative bulk of text, and c = 0.25. The product of the relative bulks which is the numerator of the above formula hits a local minima *(Edelbaum1962)* of zero whenever either of b_1 or b_2 is zero i.e. any of the two documents being compared at once during a plagiarism check

is an empty document, and it hits a global maxima *(Gan and Jiang1999)* of 0.25 when the two documents are equally bulky having same number of characters, words or any other text unit we specify in calculating bulk of the text. It is as shown in fig (1).

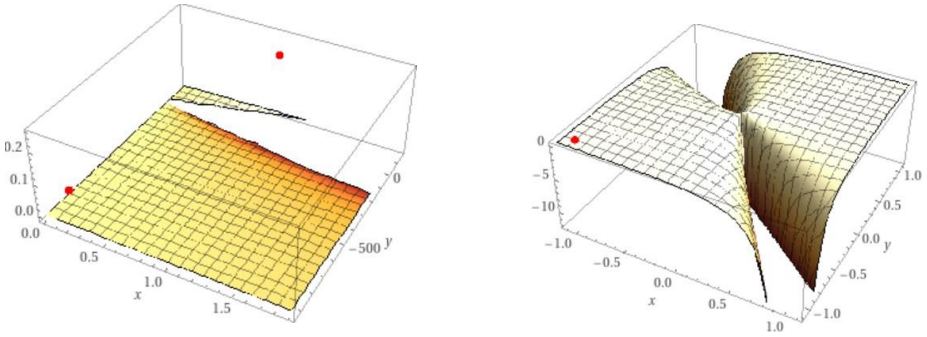


Figure 1. 3D plots showing global maxima of 0.25 found at (1,1) for domain $x \geq 0$ (on left) and at (-1,-1) for domain $x < 0$ (on right) showcasing $b1 = b2$ condition for maxima (for function $z = x\,y/\,(x+\,y)^{2}$, x representing b_1, y representing b_2)

When $b_1 = b_2$ and $d = 0$, then S = 1, which is maximum value of S

When $b_1 = 0$ or $b_2 = 0$, then S = 0, which is minimum value of S

Therefore, 0 ≤Gravity Score (S)≤ 1

III. Bulk of text and its effect on Gravity Score

Computation of bulk or quantification of how large sized a text is can be done in several ways. Depending on the definition of text token (unit of text) *(Grefenstette and Tapanainen 1994)* for a particular process, bulk for the same text document can be different. If the text token is character level *(Chang and Manning 2014)*, the bulk of the text would be the number of characters in it which is clearly higher than the count if the text token is a word or sentence. Relative bulk is however almost uniform irrespective of the text token considered. Text documents can be interpreted either statistically or semantically. Depending on the text encoding, or as termed in Natural Language Processing, text embedding technique, either the frequency of constituent words and frequency of documents with respect to contained word becomes important, or the meaning and context of the constituent words in the text get importance. Syntax in the form of context in which words are placed is given a greater weightage *(Alemi and Ginsparg2015)* by semantic text embedding techniques, while lexicons are represented well in embeddings created by both techniques. The following is to examine the effect on Gravity Score for different considerations of text tokens.

III.i. Sentence token

For the analysis, we consider *Summary of Hamlet - Act I* by Shakespeare from *shakespeare.org.uk* as the source text. We produce a copied text using a well-known open-source article rephrase software called QuillBot. The text embedding technique that is included in the comparison as representative of statistical text embedding techniques is TF-IDF (Term Frequency - Inverse Document Frequency) Vectorizer, and s-transformer is a prime semantic text embedding model. Here, analyzing source and copied text, the values relevant to our calculation are as in Table 1.

Table 1: Table showing the effect of defining text token as 'sentence' with a comparison between semantic and statistical models for text embedding. Note: d^2 has been down-scaled here by a factor of 10 for ease of threshold setting

Type of Text Embedding model	b_1	b_2	d	S
Semantic	8		1.86	0.42
Statistical			0.63	0.87

Generally, open-source article rephrasing software like Quillbot, WordTune, CleverSpinner, Chimp Rewriter, etc that are widely used for plagiarizing does not manipulate the number of sentences.

III.ii. Word token

The number of words is affected during rephrasing by this software. Therefore b_1 and b_2 for two documents are in most cases not equal when the text token is 'word'. However, the number of words in the generated text is found to be in the vicinity of that in the original text from the comparison among software as shown in fig (2).
An allied aim of article rephrasing is to reduce the count of words and thus shorten text. QuillBot, in the experiment, is found to serve the purpose best. We use the same for the remaining analysis.

Table 2: Table showing the effect of defining text token as 'word' with a comparison between semantic and statistical models for text embedding. Note: d^2 has been down-scaled here by a factor of 10 for ease of threshold setting

Type of Text Embedding model	b_1	b_2	d	S
Semantic	125	118	1.86	0.42
Statistical			0.63	0.86

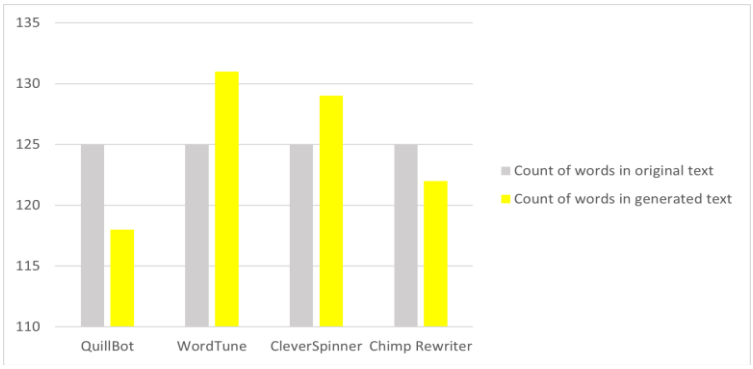
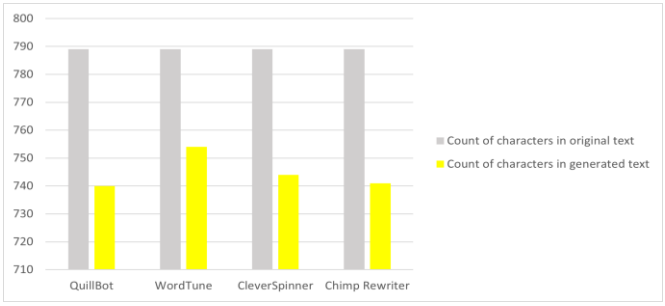


Fig. 2. Comparison of change in the number of words during document rephrasing by different software

III.iii. Character token

The number of characters that are definitely higher than the number of words in a



document is affected even more during the rephrasing of the document. From observation found in Table 1 and comparative analysis as enclosed in fig (2) and (3), it is found that in the article rephrasing for plagiarism by open source software, change in the number of text tokens depends on the number of word tokens in the original text as $|b_1 - b_2| \propto b_1$.

Fig. 3. Comparison of change in the number of characters during document rephrasing by different software

Table 3: Table showing the effect of defining text token as 'character' with a comparison between semantic and statistical models for text embedding. Note: d^2 has been down-scaled here by a factor of 10 for ease of threshold setting

Type of Text Embedding model	b_1	b_2	d	S
Semantic	789	740	1.86	0.41
Statistical			0.63	0.86

It is observed that in practical cases relative bulks of plagiarized text and source text tend to remain as close as possible to each other. This makes the bulk of text not impact a lot on Gravity score in real-world plagiarism. However, this implies that genuine documents which would tend to have the bulk of text not even close to reference documents with some same or related jargon and therefore might have low Euclidean distance will definitely have low Gravity scores and would not be accused of plagiarism. This can be demonstrated by *Summary of Hamlet* and *Summary of Hamlet - Act I* from *shakespeare.org.uk*, wherein *Summary of Hamlet* is not a plagiarized text but definitely draws reference from *Summary of Hamlet - Act I*. Its genuineness concerning the reference document is reflected in Table 4 shown below.

The above experiment also indicates a better utility of the semantic text embedding model as compared to its statistical counterpart by outputting a larger difference between Gravity Scores for

Table 4: Table showing Gravity Score of an authentic text on defining text token as 'sentence', 'word', and 'character' with a comparison between semantic and statistical models for text embedding. Note: d^2 has been down-scaled here by a factor of 10 for ease of threshold setting

Text token	Type of Text Embedding model	b_1	b_2	d	S
Sentence	Semantic	8	4	3.14	0.18
	Statistical			0.92	0.66
Word	Semantic	125	63	3.14	0.18
	Statistical			0.92	0.66
Character	Semantic	789	364	3.14	0.17
	Statistical			0.92	0.64

genuine and plagiarized texts and thus representing the truth better. This also facilitates easy threshold setting to differentiate between plagiarized and authentic texts.

IV. Dependence of Euclidean distance and Gravity Score on Text Embedding approaches

Encoding a text is done by statistical, contextual, and semantic text embedding models following approaches that are different from each other. While the total number of words in a document, the total number of documents in a corpus, the total size of the vocabulary of a corpus, the presence/absence of each word in a document, frequency of each word in a document and frequency of

documents having a particular word are important factors for statistical approach of text encoding, contextual approach encodes reflecting the relations between words. It understands relations based upon position of each word relative to its preceding or succeeding set of words. Semantic word embeddings top it by capturing not only context with respect to neighboring words but also concerning all other words in a text. These execute the process of excavating all possible contexts on already encoded versions of texts that are in form of positional embedding, sentence embedding, and token embedding.

IV.i. Statistical text embedding models

The size of text embeddings created by these models is the total number of words contained in all the documents combined. This is the size of the entire vocabulary. Each block in the array or embedding is assigned to a word. Text embedding model BOW(Bag of Words) *(Zhang, Jin, and Zhou 2010)* also known as Count Vectorizer fills each such block with the count of the corresponding word in the document concerned. Hashing Vectorizer (*Kanada 1990*) is almost the same as a Count Vectorizer except for its non-storage of vocabulary. It makes use of hashing technique to map each word to its frequency. TF-IDF(Term Frequency Inverse Document Frequency) *(Ramos* et al. *2003)* Vectorizer tops it as it fills the block with a value that represents the count of the corresponding word as well as its uniqueness for the concerned document and is calculated as

$$tfidf_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} \times \log \frac{|D|}{|dj: ti \in dj|} \tag{4}$$

where n_{ij} is the frequency of the word t_i in document d_j , $\sum_k n_{kj}$ is the count of occurrences of all words in document d_j , $|D|$ is a total number of documents in the collection of documents, and $|d_j : t_i \in d_j|$ is the count of documents having the word t_i. The size of embedding created by another approach called Co-occurrence Vectorizer *(Morita* et al. *2004)* is however not equal to the size of the vocabulary. Instead, it equals the number of distinct n-grams that can be procured from the corpus of documents, where n is predefined. This technique allots one block per n-gram for the embedding and fills it with the frequency of the corresponding n-gram as observed within the document to be encoded.

Shown below is a comparison between BOW, Count Vectorizer, and TF-IDF models concerning Euclidean distance and Gravity Score results using them. For the exercise, the data used is *Summary of Hamlet - Act V* from shakespeare.org.uk. The plagiarised text is its rephrased version produced using QuillBot. The text token considered in the analysis is 'sentence' as it is found to keep the value of relative bulk intact even on rephrasing.

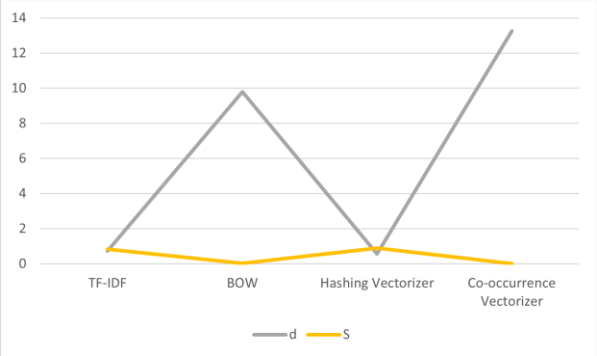


Fig.4. Comparison among Statistical text embedding models on their effect on d and S

IV.ii Contextual text embedding models

GloVe and Word2Vec are categorized as contextual embedding models as these encode each word preserving its context concerning all other words or neighboring words. GloVe (Global Vectors for Word Representation) *(Pennington, Socher, and Manning2014)* computes for each word, the probability of occurrence of each other word next to it. All these probability values are calculated for a word using the formula

$$F\ (W_i, W_j, W_k' = \frac{P_{ik}}{P_{jk}} \tag{5}$$

where $P_{i\ k}$ is the probability of occurrence of a word i next to a reference word k and $P_{j\ k}$ is the probability of occurrence of a word j next to the same reference word k , contributing to the text embedding created by GloVe. For the convenience of plotting, this embedding can be further compressed limiting the number of components of the vector. The aim of capturing contexts remains almost the same even for Word2Vec *(Mikolov* et al. *2013)*. However, the process here involves an artificial neural network(ANN) with each layer of neurons fully connected with the next layer. The system is designed to capture the context of a word concerning its neighbouring words by defining the input to the neural network as an encoded form of m^{th} n words (example: 1^{st} two words) and output as the word just following them. When the ANN trains over this pair of input and output, the weights at the last layer at the end of training act as components of the Word2Vec word embedding of the word initially fixed at the output. This serves in capturing the relation of the word concerning the words preceding it and this specific kind of WordVec model is known as CBOW(Continuous Bag of Words) Word2Vec. The neural network can also take a word as input and several words succeeding that in the output to capture the relation of the word with the terms succeeding it via the weights observed in the first layer at the end of training. This modified model is known as Skip-gram Word2Vec. Fig (5) shows the difference in approaches by CBOW and Skip-gram. All the layers in the

diagram are dense fully connected layers, with the second layer storing a hidden representation that is abstracted from both the input and output layers and is called the hidden layer. It has a high number of

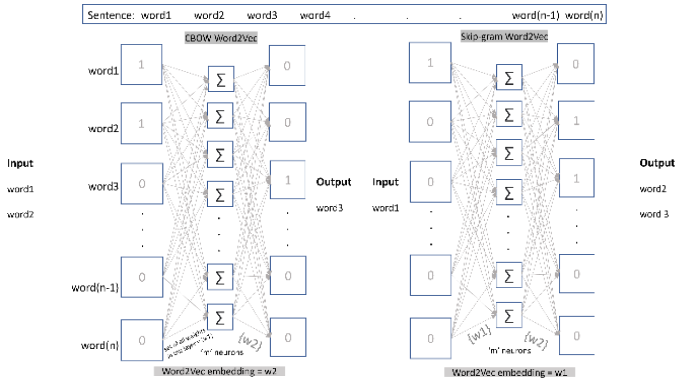


Fig.5. CBOW and Skip-gram Word2Vec models to generate contextual word embeddings

neurons. The set of weights w1 and w2 are initialized randomly and are updated during training by a backpropagation algorithm. The final states of w1, w2 contribute to text embedding. The following line chart in fig (6) is to compare contextual text embedding models concerning their impact on d and S. d^2 has again been down-scaled here by a factor of 10. GloVe and Word2Vec originally compute word embeddings. The centroid of word embeddings of all words in a document has been found to represent the entire document. The Euclidean distance between centroids for two such documents is the value of *d* that is reported here.

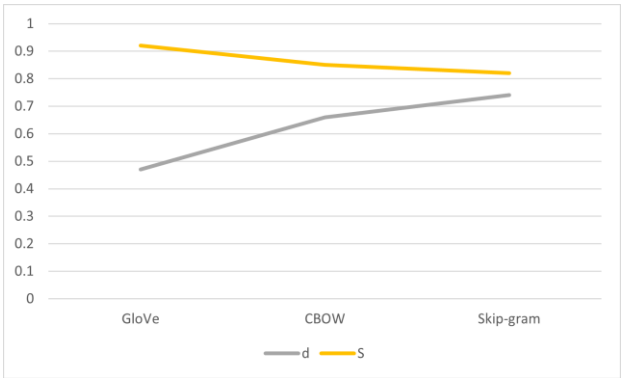


Fig.6. Comparison among Contextual text embedding models on their effect on d and S

IV.iii. Semantic text embedding models

The semantic text embedding pipeline makes use of contextual word embeddings as input and pro- cesses them to finally derive word embeddings where the meaning of each word is considered too. The transformer is the foundation of all such models. In a Transformer encoder, the initial token embed- dings where face values of words are stored by embedding models like BOW, TF-IDF, etc are added with their respective positional embeddings that represent their contexts. Positional embeddings are typically derived using contextual text embedding systems. The resultant embedding for each word is now pushed into a multi-head attention block *(Vaswani* et al. *2017)*. For this block, each word embedding is made an input as a set of three features representing weight matrices or vectors known as Key(W_k), Value(W_v), and Query(W_q). The attention that this word gets from each accompanying word in the sentence is calibrated by finding essentially a scaled dot product between their respective vectors and is formulated as

$$Z = softmax\left(\frac{W_k W_k^T}{\sqrt{Dimension\ of\ W_q\ or\ W_k or\ W_v}}\right).V \tag{6}$$

where V is the word vector for each word in the sentence. In a multi-head attention block, there are multiple Key(W_k), Value(W_v), and Query(W_q) weight matrices for each block. Therefore the output too has multiple vectors for each word which are then concatenated and multiplied with another weight matrix (W) to adjust the dimension.

$$Z' = [Z_1 Z_2 Z_3 ... Z_n].W^T \tag{7}$$

This is passed as input to a feed-forward neural network or artificial neural network (ANN) and through fake tasks like next-sentence prediction, and masked language modelling, weights procured from the output layer at the end of training serves as the semantic embedding of the entire text. This technique is put to use in language models like BERT, AlBERT(A Lite BERT), RoBERTa (Robustly Optimized BERT Pre-training Approach), etc. Semantic text embedding models yield moderately high Gravity scores and thus acknowledge the rephrasing and yet signal suspected plagiarism. In all text applications including sentiment analysis, summarization, question-answering, named entity recognition, etc, semantic models based on the Transformer pipeline have been found to achieve the best results and it would be prescribed to associate and use the merit of Gravity score as well with Transformer-based models for realistic outcomes.

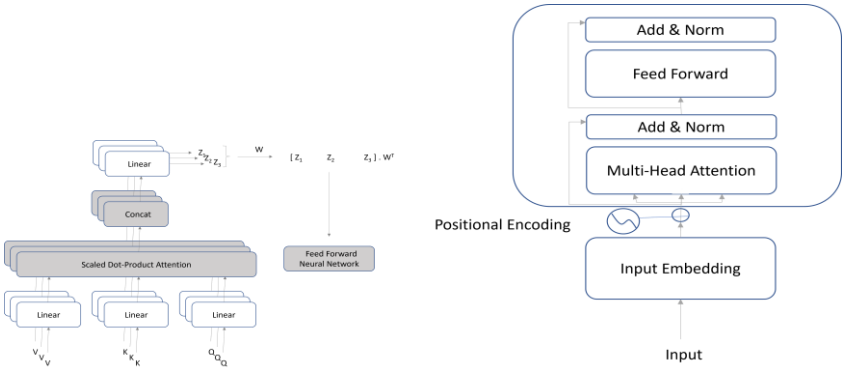


Fig.7. Multi-head attention pipeline (on left) and Transformer encoder in which the Multi-head attention block is contained (on right)

V. 2D representation of Embedding Space

Text embeddings are *n* dimensional vectors that feature represent texts. Real-world bodies can be located as points in 2D or 3D co-ordinate spaces. Similarly, text embeddings can be translated to data points and located on 2D spaces mostly by preserving their features and yet shrinking the text embeddings dimensionally using dimensionality reduction techniques like PCA(Principal Component Analysis) and t-SNE(t-distributed Stochastic Neighbour Embedding). For a corpus of documents whose size of the vocabulary is *n*, $n \in N$, TF-IDF text vector of a document has *n* components, each component representing the general importance of a word in that document. Likewise, components of text vectors or embeddings generated using other text embedding models represent other features, but the vector in almost all cases is multi-dimensional. However for visualization,

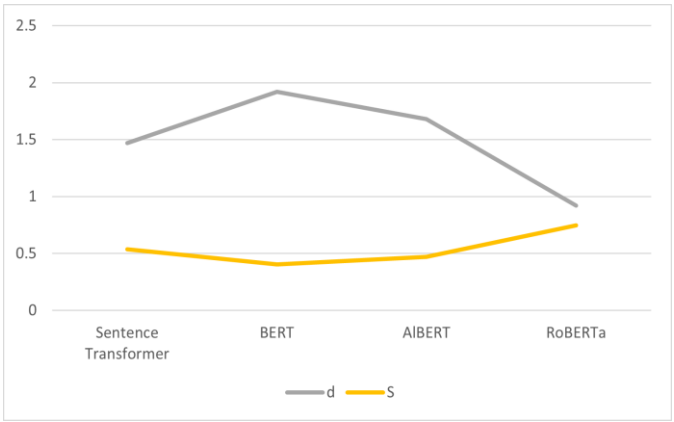


Fig.8. Comparison among Semantic text embedding models on their effect on d and S

these text vectors need to be translated into 2D embedding space. t-SNE *(Van der Maaten and Hinton 2008)* does that by comparing two probability distributions and minimizing the disparity or divergence between the two to an as low value as possible. The two probability distribution functions output the probability of each point x_j being a neighbor of each point x_i in the original dimensions and new dimensions respectively. Thus probability value for two points is proportionate to the Euclidean distances between those points and thus the probability distribution is reflective of the way the data points are oriented locally. This probability distribution is formulated as:

$$P_{j|i} = \frac{e^{\frac{-||x_j - x_j||^2}{2\sigma_i^2}}}{\sum_{i \neq k} \exp\left(\frac{-||x_i - x_k||^2}{2\sigma_i^2}\right)} \tag{8}$$

where $P_{j|i}$ is the probability of x_j and x_i being neighbors and σ^2 is the variance at each point. However, a normal distribution to which this original distribution is translated in the new dimensional system, by property of normal distribution shall have a global variance for all points. This new probability distribution is given by

$$Q_{j|i} = \frac{e^{-||y_i - y_j||^2}}{\sum_{i \neq k} e^{-||y_i - y_k||^2}} \tag{9}$$

where y_i, y_j are the data points in the new dimensional system, and for which $\sigma = \frac{1}{\sqrt{2\pi}}$ as for a normal distribution $f(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-(x-y)^2}{2\sigma^2}}$

The distribution of data points in the new set of dimensions should ideally be convergent with the original distribution to represent the same properties. This is done by minimizing the KL (Kullback-Liebler) divergence of the distribution P from the distribution Q which is given by

$$KL(P||Q) = \sum_{i \neq j} P_{ij} \log \frac{P_{ij}}{Q_{ij}} \tag{10}$$

Principal Component Analysis or PCA *(Abdi and Williams 2010)* however takes a different ap- proach of finding two principal axes that are perpendicular to each other. Text embeddings are multi-dimensional. However there is one line of best fit passing through the origin out of all possi- ble lines in all possible directions. This line in a specific direction that fits the multi-dimensional data points best is found by maximizing the sum of squares of distances of projections of these data-points from the origin. Clearly, maximizing this term is equivalent to minimizing the sum of squares of distances of the data points from the line itself ensuring that the line selected at end of the maximization is the closest possible to all data-points. Resultant line along with the direction

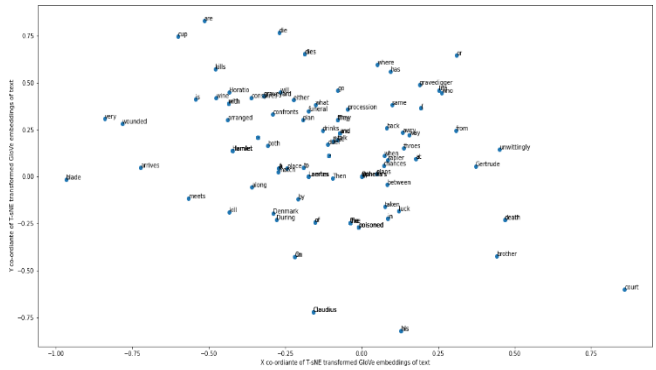


Fig. 9. t-SNE generated 2D GloVe embedding space for summary of Hamlet - Act V from shakespeare.org.uk

in which it is oriented is the first principal axis of the new set of dimensions, while the direction normal to it becomes the second principal axis. The data points which in our case are text vectors are thus now projected on a 2-dimensional screen, an example of which is shown in fig (10).

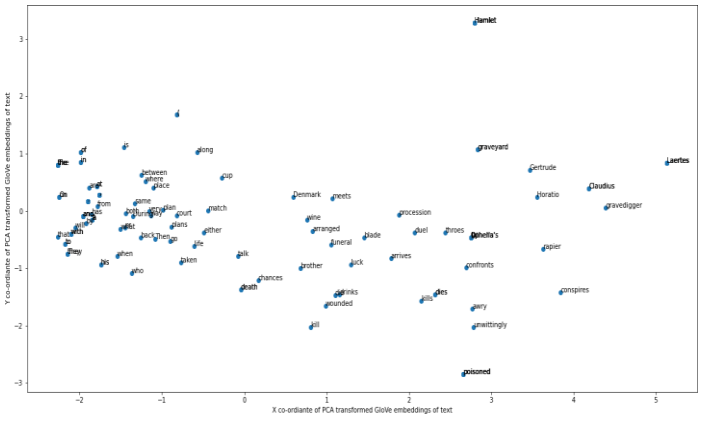


Fig.10. PCA generated 2D GloVe embedding space for summary of Hamlet - Act V from shakespeare.org.uk

VI. Comparison of Gravity Score with state-of-the-art Text similarity calculations

Plagiarism checkers make use of text similarity calculations to quantify how close a set of texts are in order to identify suspected plagiarism. In the approach, the most common text similarity or proximity measure is Cosine similarity. The wide use of the technique is due to its applicability for all types of encoding. The merit of understanding similarity in meaning or context and similarity of influence of common words that the plagiarism checker achieves based on the text embedding or encoding model used is not affected during Cosine similarity calculation. The way of calculation *(Lahitani, Permanasari, and Setiawan 2016)* is to find the cosine of the angle between two text vectors. Clearly, for smaller angles and thus closer texts, the output of cosine similarity is higher and vice-versa. If the text embedding model is semantic or contextual, higher cosine similarity indicates two texts having the nearly same meaning. Irrespective of its performance in detecting the same meanings from a set of texts, the approach is not inclusive of the size parameter for text. However, two texts that mean the same and are equally sized have more chances of the later text being copied than being derived or inspired from the source. This understanding is not used even in detecting plagiarism by Jaccard similarity *(Niwattanakul* et al. *2013)* which treats each document as a set of its constituent characters except whitespace characters and delimiters like spaces, commas, stops, etc. It measures the similarity between two documents as the ratio between the intersection of the two sets corresponding to the two documents and the union of the same two sets. Consequently, even if these measures are accurate for actually copied texts, they prove to be inaccurate for texts that are related to each other, inspired, derived, or are reference or supplementary texts and are genuine. Text similarity metrics are efficient in finding matches, but plagiarism is dependent on more factors including matches. The performance of the two approaches is enclosed in the following Table 5. For all the analysis observed throughout the paper, cosine similarity has always reported a 90+ score. This indicates its priority on reporting matches between texts, without taking into consideration whether the later text is a blind copy or is a restructured and rephrased version. The results from the Gravity score seem to recognize such possible cases and report high to moderately high values for plagiarized texts that are products of rephrasing, but not extremely high scores like Cosine similarity. Contextual text embeddings result in scores near to but still less than Cosine Similarity scores.

Table 5 : Table showing plagiarism indices reported by different approaches

Nature of document	Gravity Score	Cosine Similarity	Jaccard Similarity
Plagiarized document	0.53	0.94	0.87
Original document	0.17	0.77	0.74

The observations in table (5) are reported by taking *summary of Hamlet - Act V* from *shake- speare.org.uk* as source text, its rephrased version using Quillbot as plagiarized text, and a *summary of Hamlet* from the same website as another original text which is related to *summary of Hamlet - Act V*. The encoding considered is Sentence Transformer (semantic model) generated embedding and the text token used is 'sentence'. Here too d^2 has been down-scaled by a factor of 10 in the calculation of the Gravity score.

VII. Conclusion

There have been unquestionable improvements in the modes of detecting plagiarism and quantifying it. Manual document-to-document matching has been replaced by automated document-matching systems. The naive character-to-character matching has been succeeded by formulations on sets wherein the elements are characters of the documents. Jaccard similarity is an example and is still in use in many plagiarism checkers. However, at present, there is a major reliance on advanced natural language processing that does provide solutions in both generating rephrased articles aiming to bypass current plagiarism checkers as well as to build efficient plagiarism checkers. However a lot of drawbacks still remain, as text vectorization or embedding processes have consistently been the centre of focus and experiment and have evolved as per target, but for assessing the similarity among these encoded versions of documents, the number of similarity metrics still remains limited. There seems to be a requirement for more similarity metrics that attempt to consider all variables associated with text documents. The formulation of the Gravity score is an attempt in that direction, which understands to what degree an original document is modified or referred to, to create a new one, acknowledges the effort, and only then outputs a value that assists in deciding whether a document is plagiarized or not. It is the dependence of plagiarism checkers these days like Scribbr, Quetext, Grammarly, Unicheck, PlagScan, etc on natural language processing, that makes it almost mandatory to create trustworthy NLP models that weigh the importance of all stakeholders included in an application.

Conflict of Interest:

There was no relevant conflict of interest regarding this paper

References

I. Abdi, H., and L. J. Williams. 2010. "Principal component analysis." *Wiley interdisciplinary reviews: computational statistics* 2 (4): 433–459.

II. Alemi, A. A., and P. Ginsparg. 2015. "Text segmentation based on semantic word embeddings." *arXiv preprint arXiv:1503.05543.*

III. Cattaneo, C. 1958. "General relativity: relative standard mass, momentum, energy and gravitational field in a general system of reference." *Il Nuovo Cimento (1955-1965)* 10 (2): 318–337.

IV. Chang, A. X., and C. D. Manning. 2014. "TokensRegex: Defining cascaded regular expressions over tokens." *Stanford University Computer Science Technical Reports. CSTR* 2:2014.

V. Chujo, K., and M. Utiyama. 2005. "Understanding the Role of Text Length, Sample Size and Vocabulary Size in Determining Text Coverage." *Reading in a foreign language* 17 (1): 1–22.

VI. Danielsson, P.-E. 1980. "Euclidean distance mapping." *Computer Graphics and image processing* 14 (3): 227– 248.

VII. Edelbaum, T. N. 1962. "Theory of maxima and minima." In *Mathematics in Science and Engineering,* 5:1–32. Elsevier.

VIII. Fock, V. 2015. *The theory of space, time and gravitation.* Elsevier.

IX. Gan, L., and J. Jiang. 1999. "A test for global maximum." *Journal of the American Statistical Association* 94 (447): 847–854.

X. Grefenstette, G., and P. Tapanainen. 1994. "What is a word, what is a sentence?: problems of Tokenisation."

XI. Kanada, Y. 1990. "A Vectorization Technique of Hashing and Its Application to Several Sorting Algorithms." In *PARBASE,* 147–151.

XII. Lahitani, A. R., A. E. Permanasari, and N. A. Setiawan. 2016. "Cosine similarity to determine similarity measure: Study case in online essay assessment." In *2016 4th International Conference on Cyber and IT Service Management,* 1–6. IEEE.

XIII. Mikolov, T., K. Chen, G. Corrado, and J. Dean. 2013. "Efficient estimation of word representations in vector space." *arXiv preprint arXiv:1301.3781.*

XIV. Morita, K., E.-S. Atlam, M. Fuketra, K. Tsuda, M. Oono, and J.-i. Aoe. 2004. "Word classification and hierarchy using co-occurrence word information." *Information processing & management* 40 (6): 957–972.

XV. Nation, P., and R. Waring. 1997. "Vocabulary size, text coverage and word lists." *Vocabulary: Description, acquisition and pedagogy* 14:6–19.

XVI. Niwattanakul, S., J. Singthongchai, E. Naenudorn, and S. Wanapu. 2013. "Using of Jaccard coefficient for keywords similarity." In *Proceedings of the international multiconference of engineers and computer scientists,* 1:380–384. 6.

XVII. Pennington, J., R. Socher, and C. D. Manning. 2014. "Glove: Global vectors for word representation." In *Pro- ceedings of the 2014 conference on empirical methods in natural language processing (EMNLP),* 1532– 1543.

XVIII. Ramos, J., *et al.* 2003. "Using tf-idf to determine word relevance in document queries." In *Proceedings of the first instructional conference on machine learning,* 242:29–48. 1. New Jersey, USA.

XIX. Van der Maaten, L., and G. Hinton. 2008. "Visualizing data using t-SNE." *Journal of machine learning research* 9 (11).

XX. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. "Attention is all you need." *Advances in neural information processing systems* 30.

XXI. Verlinde, E. 2011. "On the origin of gravity and the laws of Newton." *Journal of High Energy Physics* 2011 (4): 1–27.

XXII. Zhang, Y., R. Jin, and Z.-H. Zhou. 2010. "Understanding bag-of-words model: a statistical framework." *Interna- tional journal of machine learning and cybernetics* 1 (1): 43–52.