# Plagiarism Detection Using FP-Growth Algorithm

*Varun Nandu*
*(nandu.v@husky.neu.edu)*
*Suraj Nair*
*(nair.sur@husky.neu.edu)*

Supervised by
Dr. Lu Wang

December 8, 2017

# 1    Introduction and Related Work

Plagiarism is the "wrongful appropriation" and "stealing and publication" of another author's "language, thoughts, ideas, or expressions" and the representation of them as one's own original work. Plagiarism is considered academic dishonesty and a breach of journalistic ethics. This is a growing problem in todays times in education institutions. Thus in order to protect one's academic integrity plagiarism detection has gained a lot of importance these days. Previously works such as Plagiarism Detection Using the Levenshtein Distance and Smith-Waterman Algorithm [2] where the Levenshtein distance between two strings is given by the minimum number of operations, and that needed to transform one string into the other, where an operation is an insertion, deletion, or substitution of a single character is seen. We also came across Intrinsic Plagiarism Detection Using Character n-gram [3]. This algorithm attempts to quantify the style variation within a document using character n-gram profiles and a style change function based on an appropriate dissimilarity measure originally proposed for author identification. In this project, we implement a plagiarism detection system using Frequent Pattern (FP) Growth Algorithm.

The FP-Growth Algorithm, proposed by Han [1], is an efficient and scalable method for mining the complete set of frequent patterns by pattern fragment growth, using an extended prefix-tree structure for storing compressed and crucial information about frequent patterns named frequent-pattern tree (FP-tree). In natural language processing, we can model our text data into vectors using word vector vector representation techniques. Many word vector representation techniques have been developed in recent times. We will use GloVe [8] to represent our textual data in form of vectors. Thus we get a vector representing each unique word in our data.

For getting similar words in context and meaning, many methods have been used like Brown Clustering Algorithm [4] or Word2Vec[5]. However in our project, we use K-means [10] Clustering Algorithm because of its features like completeness, exclusivity and fast execution. In our project, clusters of similar words are created. Then a vector for each sentence is built based on those clusters. If the two words belong to the same cluster then, the same cluster number is applied to both the words in the vector. After that a vector for each text document is created based on sentence vectors. If two sentence vectors are similar, then we assign a unique number to those sentences. Thus we get a vector for each document which we can use as item-sets in the FP-Growth Algorithm and the number of documents will be our transactions. Once we get frequent item sets we know which documents are plagiarised from each other.

# 2    Dataset and Features

## 2.1    Dataset

In this study, the freely available Clough-Stevenson corpus [7] was applied. The corpus consists of answers to five short questions on a variety of topics in Computer Science field. The five short questions are: 1. What is inheritance in object oriented programming? 2. Explain the PageRank algorithm that is used by the Google search engine. 3. Explain the Vector Space Model that is used for Information Retrieval. 4. Explain Bayes Theorem from probability theory. 5. What is dynamic programming?

Table 1: Dataset statistics

| | |
|---|---|
| Number of questions | 5 |
| Number of answers per question | 20 |
| Number of Non or lightly plagiarised answers | 57 |
| Number of Heavily plagiarised answers | 19 |
| Mean no of words per answer | 216 |

Each question has 19 students answers and 1 wikipedia answer. We are also given which answers are plagiarised. There are four different types of answers. They are cut, light, non and heavy. Cut refers to answers that are cut, copy, pasted from wikipedia answer. Heavy refers to answers that are heavily copied from wikipedia answers, light refers to answers which are slightly copied from wikipedia answers with extreme changes to the structure of answers. Non refers to non plagiarised work. We can view this in the table given below.

## 2.2 Features

For our experiment, we have considered a word embedding named - GloVe [8]. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. The statistics of word occurrences in a corpus is the primary source of information available to all unsupervised methods for learning word representations, and although many such methods now exist, the question still remains as to how meaning is generated from these statistics, and how the resulting word vectors might represent that meaning. The GloVe model is trained on the non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus. Populating this matrix requires a single pass through the entire corpus to collect the statistics. For large corpora, this pass can be computationally expensive, but it is a one-time up-front cost. Subsequent training iterations are much faster because the number of non-zero matrix entries is typically much smaller than the total number of words in the corpus. GloVe is essentially a log-bilinear model with a weighted least-squares objective. The main intuition underlying the model is the simple observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning

## 3 Methods

### 3.1 K-means Clustering

K-means clustering is a type of unsupervised learning, which is used when we have unlabelled data (i.e., data without defined categories or groups). The goal of this algorithm is to find groups in the data, with the number of groups represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the K-means clustering algorithm are:
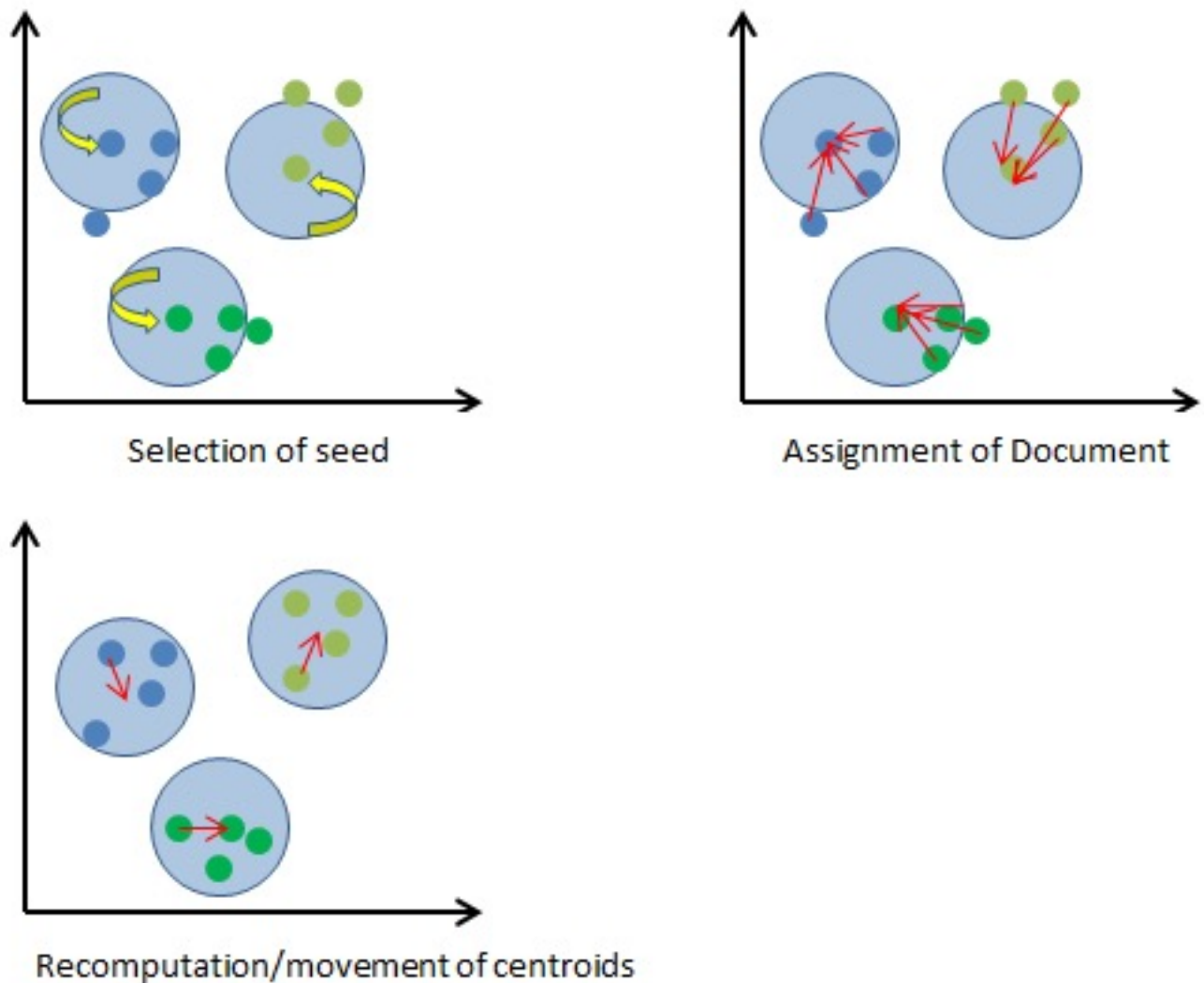
Figure 1: K-means clustering ($ImageCourtesy : DataScienceworld$[10]).

- The centroids of the K clusters, which can be used to label new data.

- Each data point is assigned to a single cluster.

Rather than defining groups before looking at the data, clustering allows you to find and analyze the groups that have formed organically. The "Choosing K" section below describes how the number of groups can be determined.Each centroid of a cluster is a collection of feature values which define the resulting groups. Examining the centroid feature weights can be used to qualitatively interpret what kind of group each cluster represents.

### 3.1.1 Algorithm

The k-means clustering algorithm uses iterative refinement to produce a final result. The algorithm inputs are the number of clusters ? and the data set. The data set is a collection of

features for each data point. The algorithms starts with initial estimates for the ? centroids, which can either be randomly generated or randomly selected from the data set. The algorithm then iterates between two steps:

1. Data assigment step:

Each centroid defines one of the clusters. In this step, each data point is assigned to its nearest centroid, based on the squared Euclidean distance. More formally, if ci is the collection of centroids in set C, then each data point x is assigned to a cluster based on

$$\arg\min_{c_i \in C} dist(c_i, x)^2$$

where dist( ) is the standard (L2) Euclidean distance. Let the set of data point assignments for each ith cluster centroid be Si.

2. Centroid update step:

In this step, the centroids are recomputed. This is done by taking the mean of all data points assigned to that centroid's cluster.

$$c_i = \frac{1}{|S_i|} \sum_{x_i \in S_i x_i}$$

The algorithm iterates between steps one and two until a stopping criteria is met (i.e., no data points change clusters, the sum of the distances is minimized, or some maximum number of iterations is reached).

This algorithm is guaranteed to converge to a result. The result may be a local optimum (i.e. not necessarily the best possible outcome), meaning that assessing more than one run of the algorithm with randomized starting centroids may give a better outcome.

### 3.1.2 Implementation

K-means clustering algorithm is that it is not guaranteed to find the most optimal cluster arrangement, if you pick the wrong starting points. One method for overcoming this is to run the algorithm a number of times with different randomly selected starting points, and then pick the solution that has the lowest total squared Euclidean distance. This approach is used in the scikit-learn package, defaulting to 10 separate repetitions. Scikit-learn implementation of K-Means returns an object that indicates the cluster to which each input vector belongs.

### 3.2 Vector Representation

From the obtained clusters and its respective labels from K-means clustering algorithm, we know use these clusters to convert our data in each documents to vectors. For this, we do the following.

- Determine the cluster label for each of the word in sentences.

- From the word vectors, we merge all the words and form sentence vectors.

- From these sentence vector, we form answer vector.

The answer vector obtained from this step will be provided as an input as transactions to our FP-growth algorithm.

### 3.3 FP-growth algorithm

The FP-growth algorithm is currently one of the fastest approaches to frequent item set mining. One of the currently fastest and most popular algorithms for frequent item set mining is the FP-growth algorithm [12]. It is based on a prefix tree representation of the given database of transactions (called an FP-tree), which can save considerable amounts of memory for storing the transactions. The basic idea of the FP-growth algorithm can be described as a recursive elimination scheme: in a preprocessing step delete all items from the transactions that are not frequent individually, i.e., do not appear in a user-specified minimum number of transactions. Then select all transactions that contain the least frequent item (least frequent among those that are frequent) and delete this item from them. Recurse to process the obtained reduced (also known as projected) database, remembering that the item sets found in the recursion share the deleted item as a prefix. On return, remove the processed item also from the database of all transactions and start over, i.e., process the second frequent item etc. In these processing steps the prefix tree, which is enhanced by links between the branches, is exploited to quickly find the transactions containing a given item and also to remove this item from the transactions after it has been processed.

## 4 Experiments and Results

As, described in Section 2, Each question has 19 students answers and 1 wikipedia answer. We calculated all the ground truth for our data.

Table 2: Evaluation Metrics

| Precision | 90% |
|-----------|-----------|
| Accuracy | 63.22% |
| Recall | 59.34% |
| F1-score | 71.52% |

The table below shows the plagiarised task and student names obtained form our algorithm.

Table 3: Plagiarism detection

| taske | orig, g0pE, g2pB, g1pB |
|-------|--------------------------------|
| taska | g0pE, g3pC, g2pC, g4pC, orig |
| taskd | g2pA, g4pC, g3pA |

## 5 Future work

In our future work, we plan on using other vector representation techniques such as word2vec for a better results. We can experiment with above proposed solutions for identifying plagia-

rism in programming environment. We need to add more features when we change the domain from documents to programming. We need to modify support in FP-growth algorithm.

# References

[1] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD?00, Dallas, TX). ACM Press, New York, NY, USA 2000

[2] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4603758

[3] http://citeseerx.ist.psu.edu/viewdoc/summary

[4] Percy Liang (2005). Semi-Supervised Learning for Natural Language (PDF) (M. Eng.). MIT. pp. 44?52

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean. *Distributed Representations of Words and Phrases and their Compositionality*, In NIPS, 2013

[6] https://en.wikipedia.org/wiki/K-meansclustering

[7] P. Clough and M. Stevenson. ?Developing a Corpus of Plagiarized Short Answers, Language Resources and Evaluation: Special Issue on Plagiarism and Authorship Analysis, In Press.? Internet: http://ir.shef.ac.uk/cloughie/resources/plagiarismcorpus.html/Download, Sep. 10, 2009 [Oct. 12, 2011].

[8] Jeffrey Pennington and Richard Socher and Christopher D. Manning. *GloVe: Global Vectors for Word Representation*, p 1532 - 1543, EMNLP 2015

[9] https://github.com/stanfordnlp/GloVe

[10] https://www.datascience.com/blog/k-means-clustering

[11] http://douglasduhaime.com/posts/clustering-semantic-vectors.html

[12] https://www.researchgate.net/profile/Christian-Borgelt/publication/234774858-keeping-things-simple-Finding-frequent-item-sets-by-recursive-elimination/links/00b7d514b21cd06b09000000.pdf