Georgia State University
CSC 8980 Topics In Computer Science
Natural Language Processing
Spring 2023

PROJECT REPORT ON

DETECTING AND VALIDATING TERM ASSOCIATIONS IN TEXT CORPUS

DONE BY:

NITIN KANKANALA (002705445)

SRINIVAS NARNE(002705961)

GEETHANJALI NAGABOINA(002705458)

ABSTRACT: The text discusses the significance of identifying characteristic and discriminating terms and their semantic relationships in text processing applications, particularly in term clustering techniques. The traditional statistical co-occurrence analysis approach considers only the immediate neighbors or sentence-level co-occurrence of two terms. However, the article proposes a flexible approach that uses co-occurrence windows of arbitrary sizes to find statistically significant correlations between two or more terms. This method captures more nuanced semantic relationships between terms and can improve the accuracy of text-processing tasks. The goal is to identify relationships between words or phrases that frequently co-occur in a text corpus and determine whether these relationships are statistically significant and meaningful. This information can then be used in various natural language processing tasks, such as text classification, information retrieval, and sentiment analysis, to improve their accuracy and effectiveness.

INTRODUCTION

In text processing applications, determining characteristic, and discriminating terms, as well as their semantic relationships is essential for various natural language processing (NLP) tasks such as text classification, information retrieval, and sentiment analysis. Cooccurrences refer to word pairs that appear together in a defined window of n words, regardless of their order. The most common types of co-occurrences are word pairs that immediately follow each other or appear together in a sentence. Traditional statistical co-occurrence analysis approaches typically consider only the immediate neighbors or sentence-level co-occurrences of two terms. However, this approach might miss the more nuanced relationships between terms that occur in a wider context.

To address this limitation, flexible approaches have been proposed to find statistically significant correlations between two or more terms using co-occurrence windows of arbitrary sizes. These approaches can capture more subtle semantic relationships between terms that may not be apparent when only considering immediate neighbors or sentence-level co-occurrences.

The proposed approach allows for identifying relationships that might not be apparent in traditional methods and can improve the accuracy and effectiveness of various NLP tasks. By utilizing co-occurrence windows of different sizes, the proposed approach can reveal more comprehensive and meaningful semantic relationships between terms. This information can then be used to improve the accuracy of various NLP tasks, such as information retrieval, text classification, and sentiment analysis.

The relationships identified between co-occurring terms using traditional statistical co-occurrence analysis approaches can be unspecific, as they only allow for statements such as "word A has something to do with word B" (and vice versa). However, several established measures exist to calculate the statistical significance of co-occurrences by assigning them a significance value. If the significance value exceeds a pre-set threshold, the co-occurrence can be considered meaningful, and a semantic relationship between the terms involved can often be inferred from it.

These measures are useful in determining the strength of the relationship between co-occurring terms, and their statistical significance is essential in determining whether the relationship is meaningful or merely coincidental. By using these measures, the proposed approach of using co-

occurrence windows which can identify not only the presence of co-occurrences but also their significance, allowing for a more accurate and meaningful understanding of the relationships between terms in text processing applications.

APPROACH

To achieve our goal of identifying term associations in a dataset consisting of 2000 Wikipedia articles, we will use various NLP techniques such as tokenization, POS tagging, and dependency parsing to extract syntactic and semantic information from the text corpus. we will use a novel approach that determines statistically significant correlations between two or more words using co-occurrence windows of arbitrary sizes. This approach will allow us to identify relationships between terms that may not be apparent in traditional approaches, leading to a more accurate and nuanced understanding of the associations between words.

Additionally, we will incorporate semantic relationships between terms using the hits algorithm, further improving the accuracy of our analysis. After constructing a co-occurrence matrix based on the extracted syntactic and semantic information from the text corpus, we can apply the hits algorithm to obtain hub scores and authority scores.

In the context of co-occurrence analysis, hub scores represent the degree to which a term appears in a cluster of highly correlated terms, while authority scores represent the degree to which a term is highly correlated with other terms in the cluster. By computing these scores, we can identify key terms in the text corpus that play a significant role in defining clusters of related terms. These scores can also be used to improve the accuracy and effectiveness of text processing applications such as information retrieval and text classification, as they provide a more nuanced understanding of the relationships between terms in the corpus. Overall, the application of the hits algorithm to co-occurrence analysis is a powerful technique for identifying key terms and improving the accuracy of NLP tasks.

After obtaining the hub and authority scores using the hits algorithm, we can further validate the statistical significance of the term associations by calculating the Pointwise Mutual Information (PMI) between the terms.

PMI is a measure of the degree to which two terms occur together more often than would be expected by chance. By comparing the observed co-occurrence frequency of two terms to the frequency that would be expected if they were independent, we can determine whether their association is statistically significant. If the PMI between two terms is greater than a certain threshold, then we can infer that the association between the terms is meaningful and can be used to inform further analysis or applications. In this way, PMI can be used to validate the results obtained from the hits algorithm and improve the accuracy of our term association detection.

Our primary approach will involve the following steps:

Data pre-processing: We will clean and pre-process the text corpus by removing stop words, and punctuation, and performing lemmatization to reduce the dimensionality of the data.

Overview of the steps involved in text pre-processing, which includes removing stop words, punctuation, and performing lemmatization:

1. **Tokenization:** Break down the text corpus into individual words or tokens.
2. **Lowercasing:** Convert all the tokens to lowercase, so that the same word is not treated as different words due to different casing.
3. **Removing Punctuation**: Remove any punctuation marks from the text corpus since they do not add any meaningful information to the text.
4. **Removing Stop Words:** Remove common stop words, such as 'the', 'is', 'and', etc., which do not carry significant meaning in the text.
5. **Lemmatization:** Convert each word to its base or root form, also known as a lemma, to reduce the dimensionality of the data.
6. **Filtering out rare words:** Remove words that occur very rarely in the text, which may not add much value to the analysis.

These steps will help in cleaning and pre-processing the text corpus to make it ready for further analysis or modelling.

Tokenization is the process of splitting a text into individual words or tokens. POS tagging, or part-of-speech tagging, is the process of assigning a grammatical tag to each word based on its context in a sentence (e.g., noun, verb, adjective, etc.). Together, these techniques can be used to extract terms or phrases from a corpus that are most relevant to a particular analysis or topic. For example, we might extract all the nouns or noun phrases from a corpus to identify the most commonly used concepts or topics.

title	main_text	name	url	datePublished	headline	nr_tokens	nr_characters
Load balancing (computing) - Wikipedia	Set of techniques to improve the distribution ...	Load balancing (computing)	https://en.wikipedia.org/wiki/Load_balancing_(...	2002-07-09T03:15:47Z	set of techniques to improve the distribution ...	6430	40840
Flock (company) - Wikipedia	Look for Flock (company) on one of Wikipedia's...	Load balancing (computing)	https://en.wikipedia.org/wiki/Load_balancing_(...	2002-07-09T03:15:47Z	set of techniques to improve the distribution ...	161	1079
Health technology - Wikipedia	(Redirected from Medical technology)Applicatio...	Health technology	https://en.wikipedia.org/wiki/Health_technology	2001-12-17T01:15:49Z	application of organized knowledge and skills ...	5932	40665
Virtual private server - Wikipedia	One of many virtual machines running on a sing...	Virtual private server	https://en.wikipedia.org/wiki/Virtual_private_...	2004-03-26T22:06:40Z	one of many virtual machines running on a sing...	877	5842
History of Asia - Wikipedia	(Redirected from Asian history)This article ne...	History of Asia	https://en.wikipedia.org/wiki/History_of_Asia	2001-11-07T20:30:57Z	history of Asia, including the continent as we...	14948	95158
...	...	...	...	...	...	...	...
Router - Wikipedia	Look up router in Wiktionary, the free diction...	Router	https://en.wikipedia.org/wiki/Router	2011-07-27T05:33:49Z	Wikimedia disambiguation page	69	495
Droid Razr - Wikipedia	(Redirected from Motorola Droid RAZR)Android s...	Droid Razr	https://en.wikipedia.org/wiki/Droid_Razr	2011-10-18T16:48:29Z	smartphone model	1791	11864
Canon EOS 80D - Wikipedia	Digital camera modelThis article needs additio...	Canon EOS 80D	https://en.wikipedia.org/wiki/Canon_EOS_80D	2016-02-18T12:53:57Z	digital camera model	1037	6426

Fig 1. The Data Frame after cleaning and pre-processing the text and contains extracted features such as the number of tokens, characters, nouns, adjectives, verbs, lemmas, named entities, noun chunks, and other grammatical constructs.

After extracting the features from the text corpus, the next step is to identify the most common nouns and noun chunks in the dataset. This can be useful in identifying the main topics or themes that appear frequently in the text.

```
In [7]: nouns.most_common()

Out[7]: [('original', 28530),
         ('data', 12618),
         ('system', 10024),
         ('time', 8224),
         ('network', 7080),
         ('software', 6985),
         ('computer', 6380),
         ('article', 6334),
         ('information', 6301),
         ('systems', 6252),
         ('users', 6065),
         ('memory', 5830),
         ('use', 5728),
         ('number', 5613),
         ('company', 5491),
         ('user', 5276),
```

```
In [10]: noun_chunks.most_common()

Out[10]: [('the original', 28524),
          ('the company', 2510),
          ('the verge', 2270),
          ('the wayback machine', 1692
          ('the use', 1643),
          ('the united states', 1602),
          ('the world', 1369),
          ('the number', 1284),
          ('the internet', 1281),
          ('the original (pdf', 1277),
          ('the user', 1273),
          ('new york', 1147),
          ('the end', 1135),
          ('the device', 1115),
          ('the new york times', 1092)
          ('the time', 1048),
```

Fig 2. The most common Nouns and Noun_chunks in the text corpus

```
<AxesSubplot:xlabel='no_tokens', ylabel='no_sentences'>
```

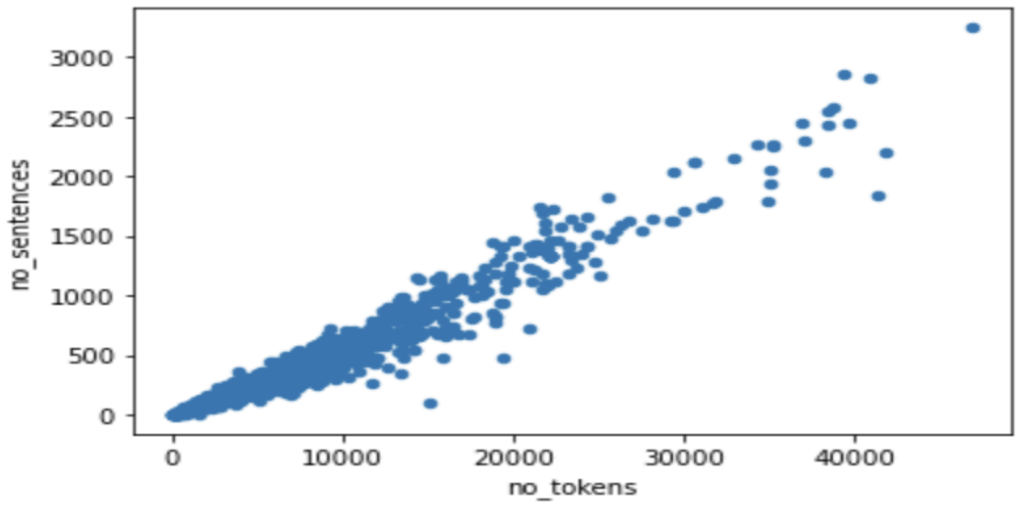

Fig 3. The resulting plot can provide insights into the distribution of sentence lengths and token counts in the dataset.

```
c["data"].most_common()
```

```
('storage', 391),
('information', 352),
('user', 343),
('network', 342),
('memory', 319),
('access', 309),
('computer', 286),
('users', 276),
('analysis', 263),
('rate', 261),
('system', 255),
('mining', 251),
('time', 237),
('database', 232),
('original', 224),
('processing', 214),
('structure', 211),
('devices', 210),
('structures', 207),
('types', 205),
('use', 204),
```

```
c["system"].most_common()
```

```
[('operating', 1593),
 ('system', 976),
 ('computer', 362),
 ('software', 346),
 ('file', 298),
 ('systems', 277),
 ('data', 255),
 ('memory', 235),
 ('information', 229),
 ('user', 218),
 ('version', 205),
 ('control', 202),
 ('time', 200),
 ('original', 195),
 ('users', 183),
 ('use', 174),
 ('security', 170),
 ('network', 162),
 ('chip', 157),
 ('management', 156),
```

Fig 4. The dictionary where each key is a word that co-occurs with "data", and "system" and each value is the frequency count of the co-occurrence. Finally, the code sorts the dictionary by frequency count and prints the top most frequent co-occurring words.

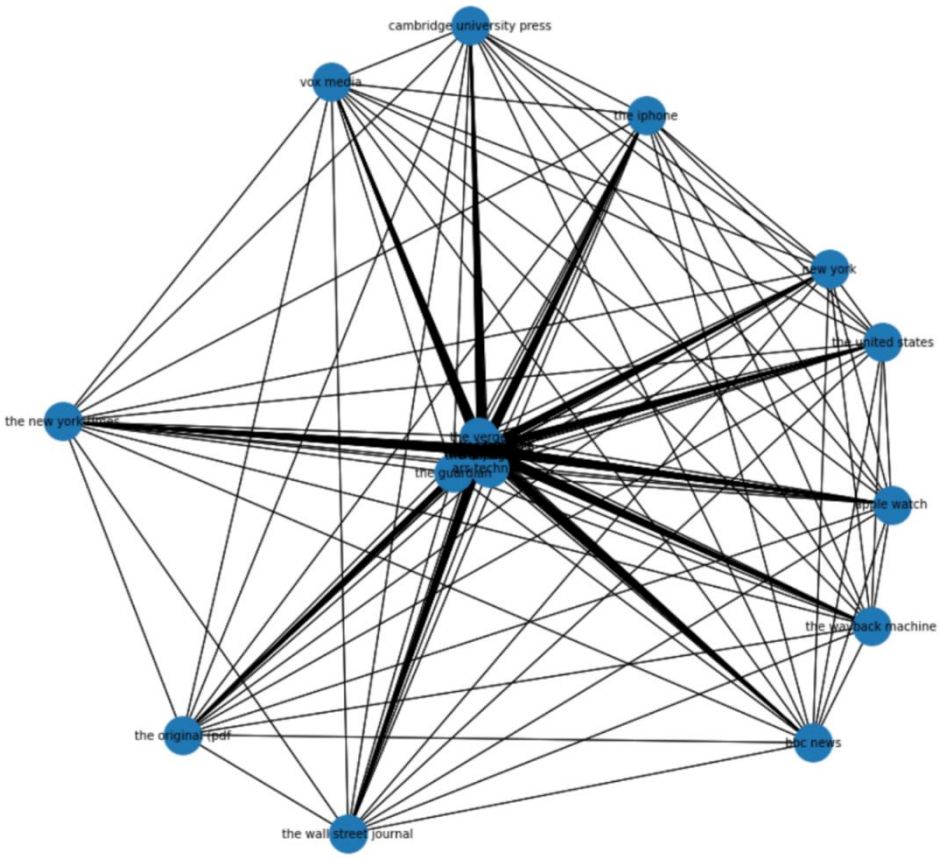

Fig 5. *A Co-occurrence graph for the topmost frequent co-occurring words*

The resulting graph shown above is a visualization of a directed graph where each node represents a noun chunk, and each directed edge represents the co-occurrence frequency between two noun chunks. The thickness of the edge represents the strength of the co-occurrence relationship. By analyzing this graph, we can identify the most frequent noun chunks and the relationships between them based on their co-occurrence patterns in the text data.

CO-OCCURRENCE MATRIX

A co-occurrence matrix is a table that shows the frequency with which pairs of items occur together in a given context. In the context of natural language processing, a co-occurrence matrix can be used to represent the frequency with which pairs of words appear together in a corpus of text. The matrix is typically a square matrix where each row and column corresponds to a unique word in the vocabulary of the corpus. The value in each cell of the matrix represents the number of times the words in the corresponding row and column co-occur within a certain window size in the corpus.

The idea behind the co-occurrence matrix is still the same. We will count the co-occurrence of words within a certain window size. Let's assume we have the following sentence:

"**I am learning natural language processing[1] using Python and I find it very interesting."**

We will preprocess this sentence by lowercasing, removing punctuation, and tokenizing it:

"i am learning natural language processing using python and i find it very interesting"

Next, we will define a window size of 9, which means we will count the co-occurrence of words within a window of 9 words. We will use a sliding window approach, where we slide the window of 9 words over the sentence, counting the co-occurrence of words within the window.

The first window will be "i am learning natural language processing using". We will count the co-occurrence of words within this window and update the co-occurrence matrix accordingly. The second window will be "am learning natural language processing using python and". Again, we will count the co-occurrence of words within this window and update the co-occurrence matrix.

A larger window size, such as 9, can capture more long-range dependencies between words in the text. This can be useful in some applications where understanding the context of a word over a longer span is important. For example, in natural language processing tasks such as sentiment analysis or named entity recognition, a larger window size can capture more meaningful associations between words and help improve the accuracy of the model.

We will continue sliding the window over the sentence, updating the co-occurrence matrix for each window until we reach the end of the sentence. The resulting co-occurrence matrix will have the same dimensions as the vocabulary, where each row and column correspond to a word in the vocabulary. The value in the (i, j) position of the matrix will represent the number of times word i co-occurs with word j within a window of 9 words.

We can use this co-occurrence matrix for various natural language processing tasks, such as text classification, information retrieval, and word embeddings.

This code snippet extracts terms from the pre-processed corpus by counting co-occurrences of words within a sliding window of size window_size (here 9) over the corpus. It creates two defaultdict objects, word_cnts_in and word_cnts_out, that count the frequency of words occurring within the window before and after the current word respectively. It then fills in two co-occurrence matrices, cooccurrence_matrix_in and cooccurrence_matrix_out, where each row and column represent a word in the vocabulary, and each entry represents the number of times a word co-occurs with another word in the vocabulary within the specified window. The voc variable is assumed to contain the vocabulary of words extracted from the corpus.

HITS ALGORITHM

HITS algorithm, which stands for "Hyperlink-Induced Topic Search." It is a ranking algorithm that is used to rank web pages based on their relevance to a given search query. In this case, we are using the algorithm to rank words in our co-occurrence matrix based on their authority and hub scores. The algorithm starts by initializing authority and hub scores to 1 for each word in the vocabulary. Then, for a fixed number of iterations (in this case, 11), it updates the authority scores and hub scores based on the co-occurrence matrix.

The co-occurrence matrix is a weighted, directed graph, where the nodes are words and the edges represent co-occurrences between words. The weight of an edge represents the frequency of co-occurrence between the two words. To update the authority scores, the algorithm multiplies the co-occurrence matrix by the hub scores and normalizes the result. To update the hub scores, the algorithm multiplies the co-occurrence matrix by the authority scores and normalizes the result.

The normalization is necessary to ensure that the scores sum to 1. After each iteration, the algorithm checks for convergence by calculating the difference between the new and old authority and hub scores. If the difference is smaller than a tolerance value (in this case, 1e-6), the algorithm terminates early. Finally, the algorithm ranks the words based on their authority and hub scores. The words with the highest authority scores are the ones that are frequently referred to by other words with high hub scores, while the words with the highest hub scores are the ones that frequently refer to other words with high authority scores.

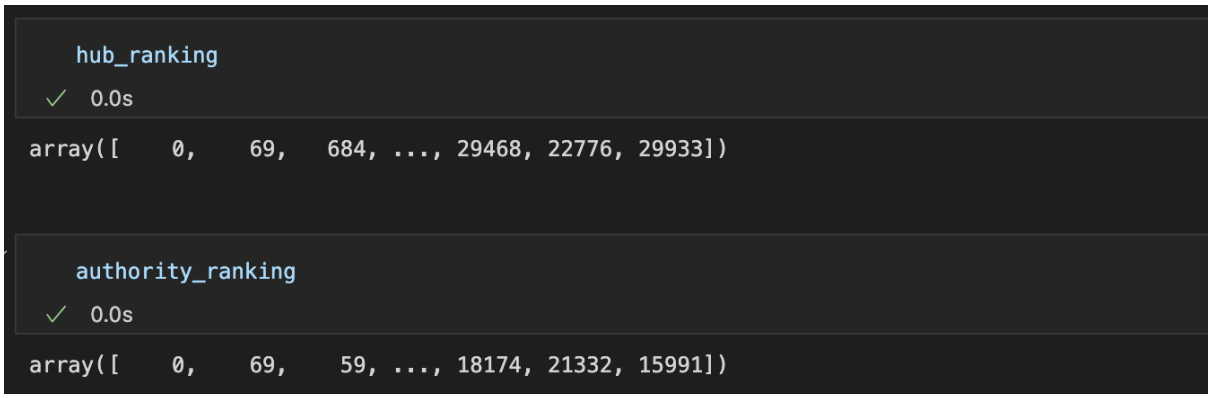

Fig 6. Hub_ranking and Authority_ranking words ranking based on authority and hub scores

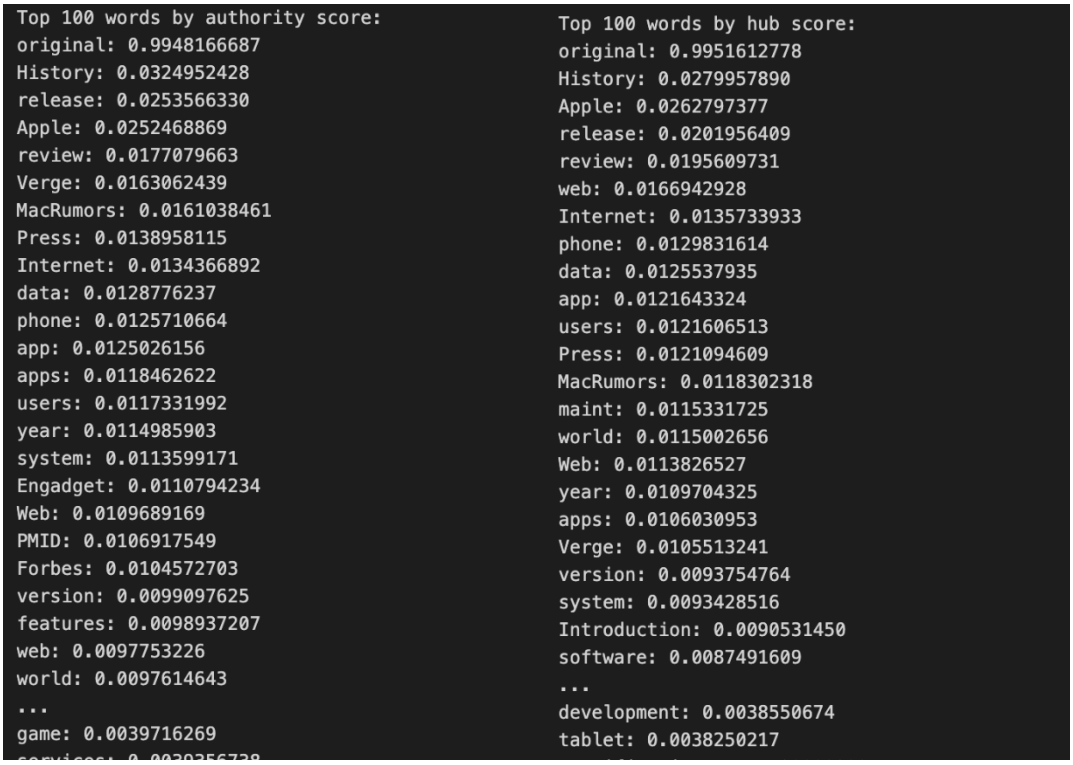

Fig 7. The top 100 words with high authority and hub scores

SEMANTIC MEANING USING WORDNET

The next step is to analyze the semantic relationships between a list of top-k words. To find the most semantically related words for each top word, we can use the WordNet corpus from the Natural Language Toolkit (nltk) library. WordNet is a large lexical database that provides semantic relationships between words, including synonyms, antonyms, and hypernyms/hyponyms. A synset is a set of synonyms that represent different senses or meanings of a word. For example, the word "bank" can have different meanings such as "financial institution" or "river bank", and WordNet provides synsets for each meaning. Once we have the synsets for a word, we can find the most semantically related word by selecting the first lemma (root form) in the first synset. A lemma is a unique concept in WordNet that represents a specific meaning of a word. For example, the lemma "bank" in the synset for "financial institution" would be "bank.n.01".We can then store the top word and its most semantically related word in a dictionary or any data structure of our choice. Overall, using WordNet to find the most semantically related words for each top word can provide insights into the relationships between different concepts in a text corpus.

The wordnet module from the Natural Language Toolkit (nltk) is used to access the WordNet database. If a synset can be found for a top word, the code retrieves the first lemma associated with the first synset and stores it as the most semantically related word for that top word in a dictionary.

```
original --> master
History --> history
Apple --> apple
release --> release
review --> reappraisal
web --> web
Internet --> internet
phone --> telephone
data --> data
users --> user
Press --> imperativeness
world --> universe
```

Fig 8. Top words and their most semantically related words

```
original:
        History (0.0323380073)
        release (0.0252339393)
        Apple (0.0251247242)
        review (0.0176222824)
        Verge (0.0162273425)
        MacRumors (0.0160259240)
        Press (0.0138285735)
        Internet (0.0133716728)
        data (0.0128153125)
        phone (0.0125102385)
data:
        original (0.0124887230)
        History (0.0004079386)
        release (0.0003183219)
        Apple (0.0003169442)
        review (0.0002223022)
        Verge (0.0002047052)
        MacRumors (0.0002021644)
        Press (0.0001744451)
        Internet (0.0001686814)
        phone (0.0001578146)
system:
        original (0.0092944245)
        History (0.0003035982)
```

Fig 9. Top 10 most semantically related words for each top word, sorted by their weights in descending order.

This above figure represents the Final Data Structure using the Authority and Hub scores. The graph is a dictionary where the keys are the words in the vocabulary, and the values are lists of the top 10 most related words along with their weight (calculated by multiplying the authority score of the related word by the hub score of the original word). The weight determines the strength of the relationship between the words, and the top 10 most related words are selected based on this weight in descending order.

The weight of an edge is determined by multiplying the authority score of the target node (j) by the hub score of the source node (i). This value represents the importance of the target node in the context of the source node.

weight = authority_scores[j] * hub_scores[i]

calculates the weight of the edge connecting node i to node j, and appends it to a list called "neighbors". The list stores the indices of the neighboring nodes along with their corresponding weights. Overall, computing the weights and selecting the top neighbors for each node is an important step in building the graph and identifying the most important nodes in a text corpus.

PMI

PMI is often used to identify which words tend to co-occur more frequently than would be expected by chance. PMI is defined as the log ratio of the joint probability of two words occurring together in a corpus, divided by the product of their individual probabilities. In other words, it measures the degree to which the occurrence of one word is correlated with the occurrence of another word in a corpus while taking into account their individual frequencies.

PMI is often used to build semantic representations of words, such as word embeddings, by measuring the similarity between words based on their co-occurrence patterns in a corpus. Higher PMI values indicate that two words are more semantically related and occur together more frequently than would be expected by chance, while lower PMI values indicate that two words are less semantically related and occur together less frequently.

The PMI score is calculated as follows:

1. The number of times **w1** and **w2** co-occur in the **in_matrix** is calculated and stored in **N_in**.
2. The number of times **w1** and **w2** co-occur in the **out_matrix** is calculated and stored in **N_out**.
3. The total number of co-occurrences of all words in the **in_matrix** is calculated and stored in **N_in_total**.
4. The total number of co-occurrences of all words in the **out_matrix** is calculated and stored in **N_out_total**.
5. The joint probability of **w1** and **w2** co-occurring together is calculated as the sum of their co-occurrence frequencies in both matrices, divided by the total number of co-occurrences in both matrices and stored in **P_w1_w2**.
6. The individual probabilities of **w1** and **w2** are calculated as the sum of their co-occurrence frequencies in their respective matrices, divided by the total number of co-occurrences in their respective matrices, and stored in **P_w1** and **P_w2**, respectively.
7. The PMI score is calculated as the log ratio of the joint probability and the product of the individual probabilities and returned as the output of the function.

The PMI score measures the degree of association between two words, considering their individual frequencies and their co-occurrence frequencies in different contexts. Higher PMI scores indicate a stronger association between two words, while lower PMI scores indicate a weaker association.

In natural language processing, PMI scores are often used to create word embeddings or to identify semantically related words.

These steps calculate the most semantically related words for each top word, and then calculates the PMI scores for each pair of top words. This measure can be used to statistically signify how strong the associations are calculated in the HITS algorithm.

Validation Comparison

We now created two Dictionaries of the most frequently occurring words in the total Corpus and then capturing the frequently associated words calculated by Hits algorithm and the PMI measure. These two dictionaries can be used for the comparison to determine how good the Hits algorithm has captured the associations based on PMI measure.

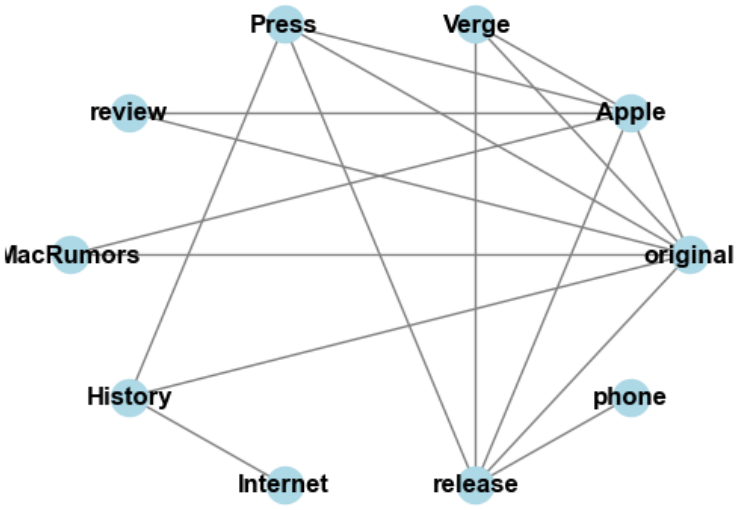

Fig 10. Common Associations of words between Hits and PMI calculated Associations.

This graph represents the common occurrence of the words say **original, History, Apple, release** with strong confidence. Strong Confidence is concluded by making the edges in above graph only if the PMI Score is good between the words for "original, History, Apple, release" in Hits Graph Data Structure.

So, say in according to Hits graph there are 10 most strongly associated words for the word Original then by the PMI calculation the above representation found out that only 7 of them are valid.

Conclusion

HITS is good at identifying the most authoritative and relevant pages in a network, which can be useful for identifying terms that are frequently mentioned together in authoritative sources. However, HITS does not take into account the strength of the relationship between terms, only the frequency of co-occurrence.

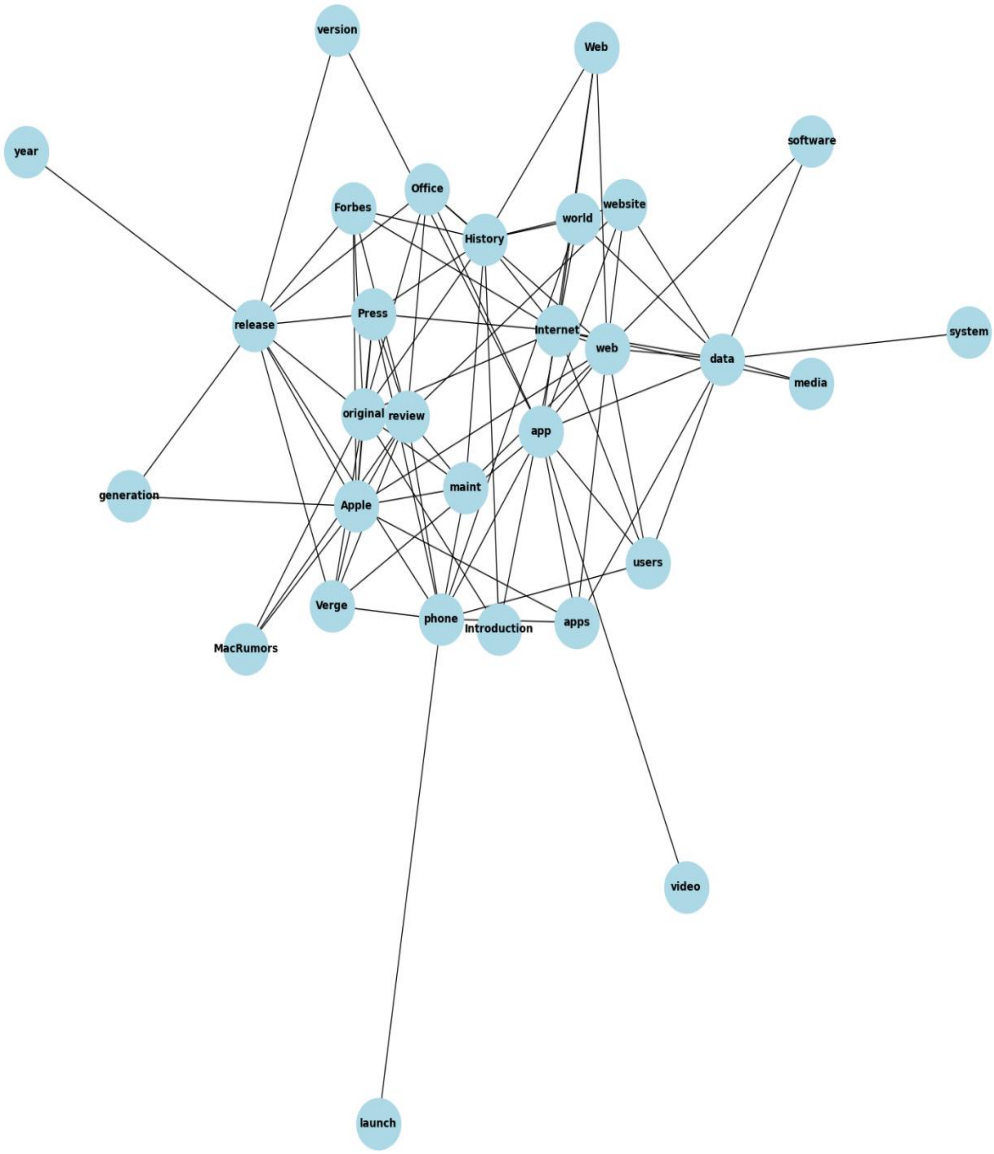

Fig 11. Directed Term Association Graph

On the other hand, PMI is good at measuring the strength of the relationship between terms, taking into account both the frequency of co-occurrence and the overall frequency of each term. However, PMI may be less effective in identifying relationships between terms that do not occur frequently together, and may not be able to distinguish between spurious and genuine associations.

By using both HITS and PMI together, we can potentially overcome these limitations and obtain a more accurate and comprehensive view of term associations. HITS can help identify potential associations, while PMI can help validate and refine these associations based on the strength of the relationship. The specific degree to which HITS is good would depend on the dataset and the specific task at hand, and would need to be evaluated empirically.

Furthermore, integrating natural language processing techniques such as topic modeling could enhance the accuracy and granularity of term associations. Overall, our study provides a foundation for future research in the field of text mining and information retrieval.

Individual Contribution

Our Team Responsibilities are as follows:

Srinivas: Extracted Data, Implemented Cooccurrence Matrix, Visualization of term Association

Geetha: Preprocessed Data, Calculated the hub and Authority Scores

Nitin: Implemented the Hits with WordNet, Validation Comparison using PMI

References

1. https://www.wikipedia.org/
2. Correlating Words – Approaches and Applications Mario M. Kubek, Herwig Unger, and Jan Dusik.
3. "Mining Association Rules in Text Documents" by Ramakrishnan Srikant and Rakesh Agrawal. This paper discusses a method for finding frequent associations between terms in text documents.
4. "Semantic Association Identification in Texts using Network Analysis" by Elham Ghasemian and Ahmad Kardan. This paper proposes a method for detecting semantic associations between terms in texts using network analysis.
5. "Statistical Analysis of Co-occurrence Patterns in Natural Language Texts" by George A. Miller. This paper discusses the use of statistical methods to detect associations between words in text corpora.
6. "Discovering Linguistic Associations with Multi-Word Expressions" by Timothy Baldwin and Su Nam Kim. This paper proposes a method for identifying multi-word expressions that are frequently associated with each other in text corpora.