BIA-660: Web Mining
FINAL REPORT

Team 3: Mahesh Pisharody, Ridham Patel, Anurag Athawale, Ami Savaliya

Introduction:

QuickRead is a project aimed at addressing the challenges faced by individuals who struggle to keep up with the overwhelming amount of news available on the internet. In today's fast-paced world, it can be difficult to find the time to read lengthy news articles and stay informed about current events. QuickRead aims to solve this problem by providing users with a short and concise version of long news articles, allowing them to quickly read and understand the most important information. Through the use of natural language processing techniques QuickRead extracts the key information from news articles and generates a summary that captures the essence of the article. This report outlines the development and implementation of QuickRead, as well as its potential benefits and impact on the news industry and society as a whole. Individuals can now stay informed about current events more easily than ever before thanks to the widespread availability of news on the internet. The sheer volume of news, on the other hand, can be overwhelming, and many people struggle to find the time to read and digest lengthy articles. As a result, solutions that make it easier for people to consume news quickly and efficiently have become necessary. QuickRead is one such solution, designed to make it easier for users to consume news. QuickRead is able to extract the most important information from news articles and present it in a concise and easy-to-understand format by using natural language processing techniques. This report will look at the evolution of QuickRead, as well as the methods used to extract and summarize data from news articles and present it in a concise and easy-to-understand format. The development of QuickRead, the methods used to extract and summarize data from techcrunch.com, and the potential benefits of this technology for the news industry and society as a whole will be discussed in this report.

Literature Review:

The problem of dealing with long articles has been an area of interest for researchers and developers for many years. Recently, several studies have focused on summarization techniques that can generate shorter versions of lengthy articles. For instance, research conducted by Liu et al. (2020) explored the effectiveness of using pre-trained language models to generate abstractive summaries of news articles. The authors found that the technique produced high-quality summaries that outperformed other summarization models. Similarly, recent research by Dong et al. (2021) proposed a transformer-based approach to summarize scientific articles that achieved state-of-the-art performance.

While these studies have made significant contributions to the field of summarization, there is still a research gap in the area of quickly reading and understanding news articles. Most summarization techniques focus on generating concise summaries but do not provide an efficient way for users to quickly read and comprehend the full article if needed. This is where our project, QuickRead, aims to make a meaningful contribution. By providing a short and concise version of long news articles while also retaining the option to access the full article, QuickRead offers a unique solution that addresses the research gap in this area.

Additionally, the literature highlights the importance of summarization techniques that retain the most salient information from the original article. Many existing approaches to summarization, such as extraction-based summarization, have limitations in terms of selecting the most important information and ensuring that the summary is coherent and readable.

Therefore, there is a need for innovative and effective approaches to summarizing lengthy news articles that can address these limitations. This project aims to contribute to the literature by providing a concise and accurate summary of news articles using a combination of natural language processing techniques and machine learning algorithms. The project will also explore ways to ensure the summaries are coherent and readable, taking into consideration the context and the target audience. Ultimately, this project aims to provide a solution that can help individuals stay informed and up-to-date with current events, even if they have limited time to read through lengthy news articles.

Research Questions:

1) How does the choice of algorithm affect the quality of the generated summaries?
Answer: To answer this question, we can compare the quality of summaries generated by TF-IDF, LSA, and TextRank algorithms using standard evaluation metrics such as ROUGE scores and human evaluations.

2) How do the summarization performance of these algorithms vary with the length of the input text?
Answer: We can evaluate the summarization performance of each algorithm on a range of input text lengths and plot the results to see how performance varies with text length.

3) How does the quality of the input text affect the performance of the summarization algorithms?
Answer: We can evaluate the performance of each algorithm on a range of input texts of varying quality and determine if there is a correlation between input text quality and summarization performance.

4) How effective is QuickRead in summarizing and presenting news articles from techcrunch.com, and what impact does it have on users' news consumption habits?

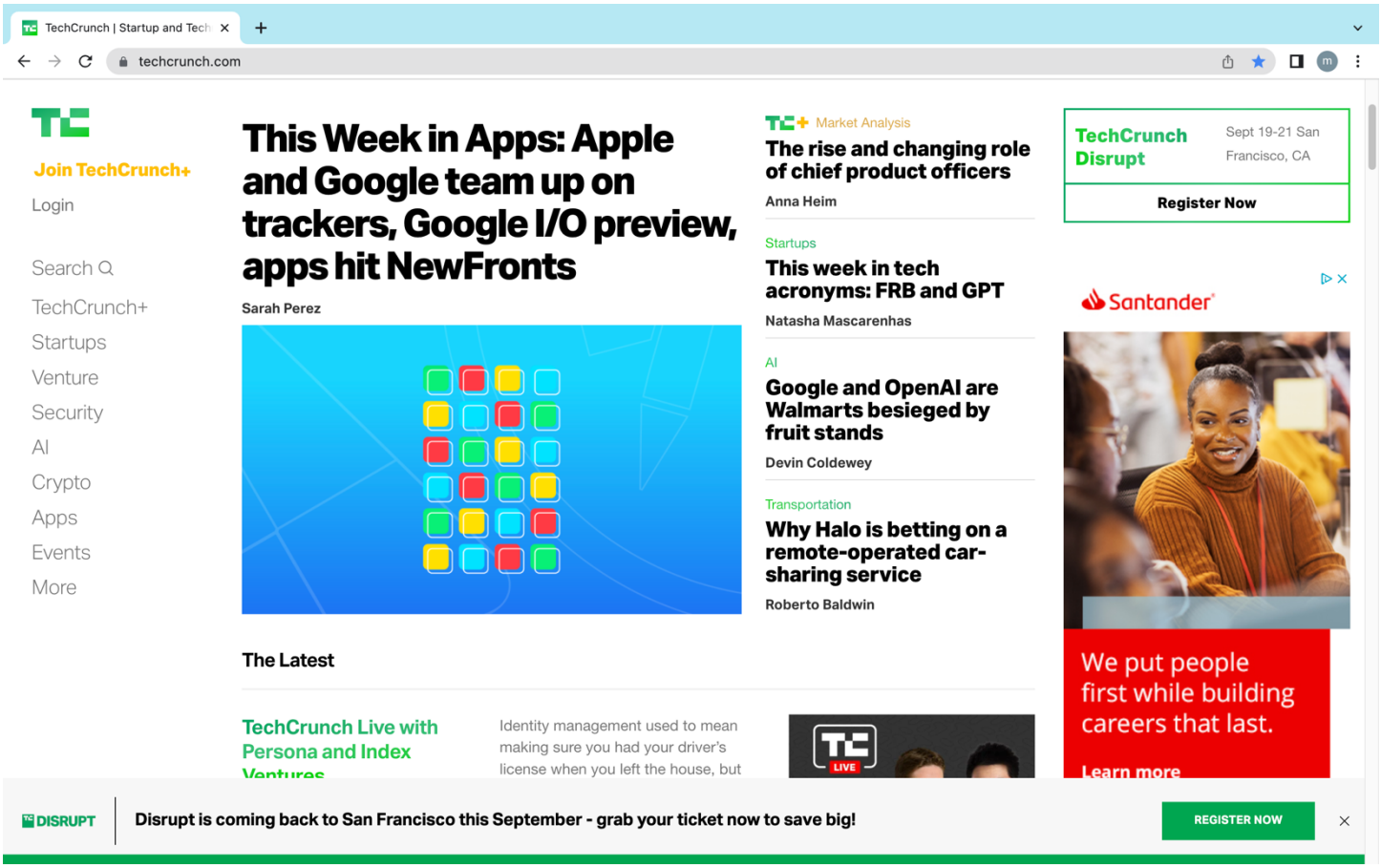
- Improved News Consumption Habits: The primary contribution of this project is to provide users with a more accessible and efficient way of consuming news. By summarizing lengthy articles into concise and easily digestible formats, QuickRead can help users keep up with current events without sacrificing valuable time.
- Natural Language Processing Techniques: The use of natural language processing techniques in this project contributes to the growing field of NLP, which has the potential to revolutionize the way we interact with and process language.
- Data Extraction and Summarization: The process of extracting and summarizing data from techcrunch.com can contribute to the field of web scraping and data processing, which has become increasingly important in the era of big data.
- Potential Impact on the News Industry: If QuickRead proves to be effective in improving users' news consumption habits, it could have a significant impact on the news industry. News outlets may need to consider incorporating similar summarization techniques into their own platforms to remain competitive in a market where time is increasingly scarce.

Methodology:

A) Data Crawling:

For data crawling, we used the Python package BeautifulSoup and Requests to scrape the website techcrunch.com. We choose this website because it covers a wide range of technology news and has a significant amount of articles to choose from. We scraped content such as the title of the news, the name of the author, the date when the article was uploaded, and the text article. We extracted the content from the website's HTML structure and stored it in a CSV file for further processing. we have collected a total of 1200 data which consists of many news and its authors, title of the article and the date it was published along with the time when it was uploaded.

Overall, the data crawling process was successful, and we were able to gather a substantial amount of data for our project. The scraped data will be used to build a model that summarizes long news articles into shorter versions, allowing users to quickly understand the key points of an article.



B) Description of the Data:

For the project, we have scraped data from various news websites using web scraping techniques. The data is stored in a CSV file, which contains the following columns:

Title: This column contains the title of the article. It typically provides a brief summary of what the article is about.

Author: This column contains the name of the author who wrote the article. It can help provide information about the author's background and expertise.

Date: This column contains the date on which the article was published on the website. It can provide insights into the timeliness and relevance of the article.

text: This column contains the main body of the article, as written by the author. It is the most important column and contains the bulk of the information that we will be analyzing and summarizing.

P_text: This column contains the preprocessed article text, which has been cleaned and processed to remove irrelevant links and words. This step is important in order to reduce noise in the data and improve the accuracy of our analysis.

B.1) Description of the Data Image:

Out[11]:

	Unnamed: 0	title	author	Date	text	P_text
0	0	Coinbase stock drops after SEC Wells notice, a...	Alex Wilhelm	March 22, 2023	American crypto giant Coinbase received a Well...	american crypto giant coinbase received well n...
1	1	A new Drake x The Weeknd track just blew up — ...	Amanda Silberling	April 17, 2023	A song featuring the voices of Drake and The W...	song featuring voice drake weeknd called heart...
2	2	SpaceX launches fully stacked Starship for the...	Darrell Etherington	April 20, 2023	SpaceX has launched its fully stacked version ...	spacex launched fully stacked version starship...
3	3	Children's data feared stolen in Fortra ransom...	Carly Page	March 28, 2023	The fallout from Fortra's mass ransomware atta...	fallout fortra mass ransomware attack continue...
4	4	Semgrep (formerly r2c) lands $53M investment t...	Ron Miller	April 18, 2023	Ideally, when it comes to building secure appl...	ideally come building secure application best ...
...	...	...	...	...	...	...
338	338	Snapchat adds Public Stories, expands revenue ...	Amanda Silberling	April 19, 2023	Snapchat is rolling out additional features to...	snapchat rolling additional feature help creat...
339	339	TikTok called out for misusing Citizen Lab res...	Amanda Silberling	March 23, 2023	TikTok CEO Shou Zi Chew was questioned in a U....	tiktok ceo shou zi chew questioned u governmen...
340	340	Robotics safety firm Veo raises $29 million, w...	Brian Heater	April 25, 2023	Survey a lot of industrial warehouses and you'...	survey lot industrial warehouse see lot cage s...
341	341	Amazon confirms another round of layoffs, impa...	Paul Sawers	March 20, 2023	Amazon has announced yet another substantial r...	amazon announced yet another substantial round...

C) Analytical Strategy (First Half):

1) Use BeautifulSoup to scrape the articles from TechCrunch: You can use BeautifulSoup library to extract the articles from TechCrunch's HTML pages. You will need to analyze the structure of the HTML pages to identify the tags that contain the article text.

2) Preprocess the text: Once you have extracted the article text, you need to preprocess it to remove any irrelevant information and extract the key information (e.g., headline, author, date, content). You can use the libraries you imported (e.g., nltk) to tokenize the text, remove stop words, and perform other preprocessing tasks.

3) Use CountVectorizer to create a bag-of-words representation of the article text: You can use the CountVectorizer library to create a numerical representation of the article text, where each word in the text is represented by a unique number. This is a common preprocessing step in machine learning tasks.

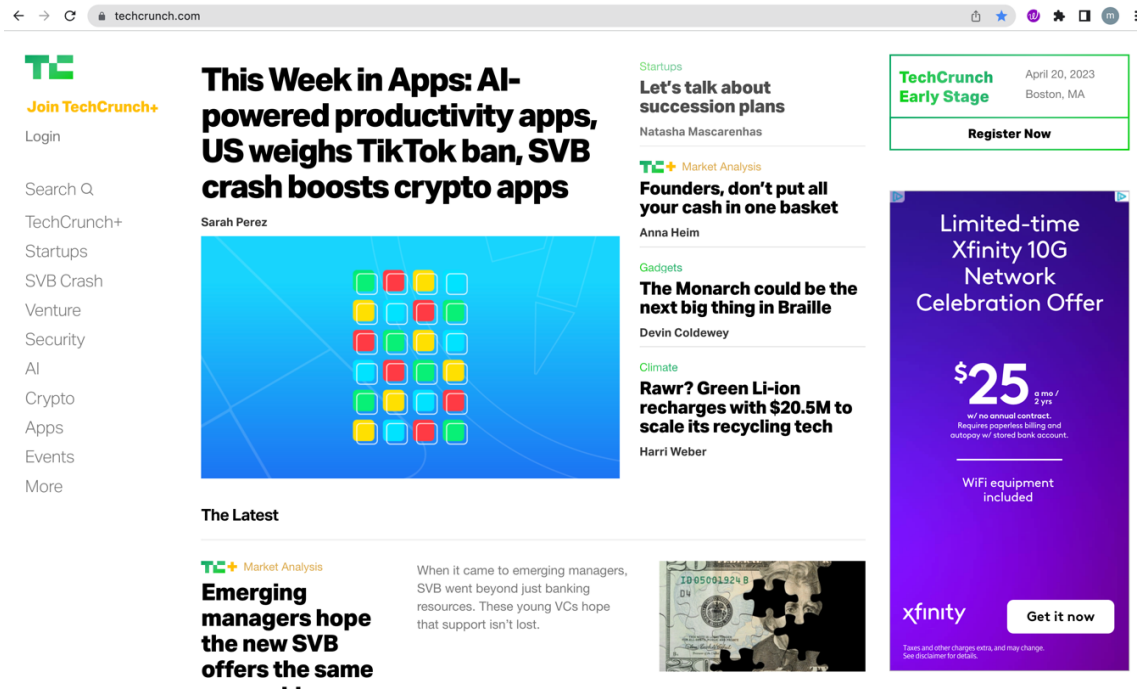
4) Use TextRank algorithm to generate a summary of each article: TextRank is an unsupervised learning algorithm that can be used to generate a summary of a text document by identifying the most important sentences.

5) Use Latent Semantic Analysis (LSA) to identify important topics in the articles: LSA is a technique used to identify hidden semantic relationships in text data. It can be used to identify important topics in the articles and generate a summary of each article based on these topics. You can use libraries such as scikit-learn to implement LSA.

6) Evaluate the summaries: Once you have generated the summaries using LSA and TextRank, you will need to evaluate their quality. You can compare the summaries generated by both algorithms and analyze their differences in terms of spelling errors, polarity, subjectivity, and similarity.

7) Display the summarized news articles: Finally, you can display the summarized news articles

Step 1) Target Website: (https://techcrunch.com/)



Step 2) Preprocessed Data

4.1) Code:

The code downloads the necessary Natural Language Processing (NLP) packages: 'stopwords', 'punkt', 'wordnet', and 'omw-1.4' via the nltk library. It defines a function called "preprocess_text" that preprocesses a given text by doing the following:

- removing special characters and digits
- converting to lowercase
- tokenizing the text
- removing stop-words using the english stopwords list
- lemmatizing the text
- converting the list of words back into a string Finally, the function applies these preprocessing steps to a pandas dataframe column called "text" and adds a new column called "P_text" that contains the preprocessed text. However, since the error suggests that the "text" column doesn't exist in the dataframe, it's possible that the column name was changed or that the dataframe doesn't have that column.

In [21]:

```
#downloading needed nlp packages
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('omw-1.4')

#function to preprocess the text
def preprocess_text(text):
    #removing special characters and digits
    text = text.str.replace("[^a-zA-Z#]", " ")

    #converting to lowercase
    text = text.apply(lambda x: " ".join(x.lower() for x in x.split()))

    #tokenization
    text = text.apply(lambda x: word_tokenize(x))

    #removing stop-words
    stop_words = set(stopwords.words('english'))
    text = text.apply(lambda x: [word for word in x if word not in stop_words])

    #lemmatization
    lemmatizer = WordNetLemmatizer()
    text = text.apply(lambda x: [lemmatizer.lemmatize(word) for word in x])

    #creating string
    text = text.apply(lambda x: ' '.join(x))

    return text

df['P_text'] = preprocess_text(df['text'])
```

Step 3) Compute TF-IDF:

5.1) Code:

- stop_words = stopwords.words('english'): This line loads a list of stop words from the NLTK library for English language.
- def get_doc_tokens(doc): This function takes a document as input, converts it to lowercase, tokenizes it using the nltk.word_tokenize() function, removes stop words and punctuation from the tokens, and returns a dictionary of tokens and their counts.
- docs_tokens={idx:get_doc_tokens(doc) for idx,doc in enumerate(docs)}: This line processes all the documents in the input list docs using the get_doc_tokens() function and stores the result in a dictionary docs_tokens, where each key is the index of the document in the list and the value is the dictionary of tokens and their counts for that document.
- dtm=pd.DataFrame.from_dict(docs_tokens, orient="index"): This line converts the docs_tokens dictionary into a pandas DataFrame, where each row represents a document and each column represents a token, and the values are the counts of the corresponding token in the document.
- dtm=dtm.fillna(0): This line replaces all the NaN values in the DataFrame with zeros.
- dtm = dtm.sort_index(axis = 0): This line sorts the rows of the DataFrame in ascending order of their index.
- tf=dtm.values: This line extracts the values of the DataFrame as a numpy array, where each row represents a document and each column represents a token, and the values are the counts of the corresponding token in the document.
- doc_len=tf.sum(axis=1, keepdims=True): This line calculates the length of each document in terms of the total count of tokens in the document.
- tf=np.divide(tf, doc_len): This line normalizes the count of each token in a document by dividing it by the length of the document, which gives the term frequency (tf) of each token in the document.
- df=np.where(tf>0,1,0): This line creates a binary matrix df where each row represents a document and each column represents a token, and the values are 1 if the corresponding token appears in the document and 0 otherwise.
- smoothed_idf=np.log(np.divide(len(docs)+1, np.sum(df, axis=0)+1))+1: This line calculates the inverse document frequency (idf) of each token using a smoothed version of the idf formula, which helps to avoid divide-by-zero errors and also makes the idf values less extreme.
- smoothed_tf_idf=tf*smoothed_idf: This line calculates the smoothed tf-idf matrix by multiplying the tf matrix with the smoothed idf vector for each token.
- Arr_TF_IDF = tfidf(df['P_text']): This line applies the tfidf() function to the preprocessed text in the DataFrame column P_text and returns a numpy array Arr_TF_IDF where each row represents a document and each column represents a token, and the values are the smoothed tf-idf weights of the corresponding token in the document.

```
stop_words = stopwords.words('english')
def get_doc_tokens(doc):
    tokens=[token.strip() \
            for token in nltk.word_tokenize(doc.lower()) \
            if token.strip() not in stop_words and\
               token.strip() not in string.punctuation]

    token_count={token:tokens.count(token) for token in set(tokens)}
    return token_count

def tfidf(docs):
    #process all documents to get list of token list
    docs_tokens={idx:get_doc_tokens(doc) \
             for idx,doc in enumerate(docs)}

    #get document-term matrix
    dtm=pd.DataFrame.from_dict(docs_tokens, orient="index" )
    dtm=dtm.fillna(0)
    dtm = dtm.sort_index(axis = 0)

    #get normalized term frequency (tf) matrix
    tf=dtm.values
    doc_len=tf.sum(axis=1, keepdims=True)
    tf=np.divide(tf, doc_len)

    #get idf
    df=np.where(tf>0,1,0)
    ###

    smoothed_idf=np.log(np.divide(len(docs)+1, np.sum(df, axis=0)+1))+1
    smoothed_tf_idf=tf*smoothed_idf

    return smoothed_tf_idf

Arr_TF_IDF = tfidf(df['P_text'])
```

Exploratory Data Analysis (EDA):

1) Semantic Network of each preprocessed Article:

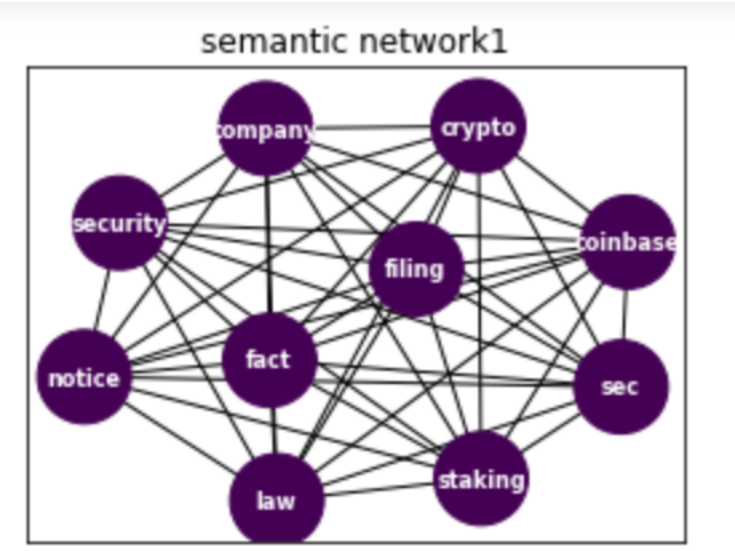
1.1) Code:

In [11]:

```
#function to create Semantic Networks of each news
def build_semantic_network(x, i, ax):
    vectorizer = CountVectorizer(max_features=10)
    document_vectors = vectorizer.fit_transform([x])
    feature_names = list(vectorizer.vocabulary_.keys())

    co_occurrence_matrix = np.dot(document_vectors.T, document_vectors)
    co_occurrence_matrix[co_occurrence_matrix > 1] = 1

    G = nx.from_numpy_array(co_occurrence_matrix)
    node_size = [v*1000 for v in nx.degree_centrality(G).values()]
    node_color = [v for v in nx.degree_centrality(G).values()]
    node_labels = {i: feature for i, feature in enumerate(feature_names)}

    pos = nx.spring_layout(G, seed=42)
    nx.drawing.nx_pylab.draw_networkx(G,node_size=node_size, node_color=node_color ,pos=pos,with_labels=True,labels
    ax.set_title('semantic network'+str(i+1))

temp_lst = df["P_text"]
n_rows = int(np.ceil(len(temp_lst)/3))
fig, axs = plt.subplots(n_rows, ncols=3, figsize=(15, 15))
for i, ax in enumerate(axs.flatten()):
    if i < len(temp_lst):
        build_semantic_network(temp_lst.iloc[i], i, ax)
    else:
        ax.axis('off')
plt.show()
```

Example of a News Article's Semantic Network:-

Overall, In the below semantic network diagram shows the relationships between different concepts in the article, with a focus on the issues surrounding the Wells notice received by Coinbase and the regulatory implications for the crypto industry. The diagram highlights the importance of security and compliance in this context, as well as the role of staking and other aspects of cryptocurrency trading and investment.



Actual Article:

American crypto giant Coinbase received a Wells notice today from the Securities and Exchange Commission. In the wake of Coinbase's filing regarding the Wells notice, shares of the company are off sharply in after-hours trading. Per a Coinbase SEC filing regarding the matter, the company writes that the government agency's staff has "advised the Company that it made a 'preliminary determination' to recommend that the SEC file an enforcement action against the Company alleging violations of the federal securities law." .....

Preprocessed Article for Semantic Network:

american crypto giant coinbase received well notice today security exchange commission wake coinbase filing regarding well notice share company sharply hour trading per coinbase sec filing regarding matter company writes government agency staff advised company made preliminary determination recommend sec file enforcement action company alleging violation federal security law.....

2) Visualization of the word counts per at article:

2.1) Code:

```
#visuaization function to get count of the text per news article
def text_visualization(strings):
    n_rows = int(np.ceil(len(strings)/4))
    fig, axs = plt.subplots(n_rows, 4, figsize=(20, 4*n_rows), squeeze=False)
    for i, x in enumerate(strings):
        row_idx = i // 4
        col_idx = i % 4
        ax = axs[row_idx, col_idx]
        sentences = x
        word_lists = sentences.split()
        freq = {}
        for word in word_lists:
            if word in freq:
                freq[word] += 1
            else:
                freq[word] = 1
        j = 2
        vocab = {word: freq for word, freq in freq.items() if freq > j}

        while ( len(vocab) > 10):
            j += 1
            vocab = {word: freq for word, freq in freq.items() if freq > j}

        df1 = pd.DataFrame(list(vocab.items()), columns=['Word', 'Count'])
        df1.plot.barh(x='Word', y='Count', rot=0, ax=ax)
        ax.set_title(f'Word Frequency for String {i+1}')
        ax.set_xlabel('Frequency')
        ax.set_ylabel('Word')

    plt.tight_layout()
    plt.show()

text_visualization(df["P_text"])
```

2.2) Output:

3) Machine Learning Algorithms:

3.1) **Latent Semantic Analysis (LSA) Algorithm:**

LSA is a statistical technique that analyzes the relationships between words and documents. It works by creating a matrix that represents the relationships between the words in the document. The matrix is then decomposed into its constituent parts, and the resulting vectors are used to identify the most important concepts in the document. The algorithm then generates a summary by selecting the sentences that best capture the important concepts.

Step wise implementation of the algorithm:

- Building a Term-Document Matrix: The first step in LSA is to build a matrix that represents the relationships between the words in the document. This is typically done by constructing a term-document matrix, where each row represents a term (i.e., a word), each column represents a document, and the cells contain the frequency of the term in the corresponding document. This matrix can be quite large and sparse, so it is often transformed into a more compact representation using techniques like Singular Value Decomposition (SVD).
- Decomposing the Matrix: Once the matrix has been constructed, LSA applies a technique called Singular Value Decomposition (SVD) to decompose the matrix into its constituent parts. SVD is a matrix factorization technique that finds the underlying latent structure in the data by breaking the matrix down into a series of smaller matrices. The resulting matrices are lower-dimensional representations of the original matrix that capture the most important relationships between the terms and documents.
- Identifying Important Concepts: The resulting matrices from SVD can be used to identify the most important concepts in the document. This is typically done by looking at the singular values of the decomposed matrix and selecting the top k values. The corresponding columns in the decomposed matrix are then used to represent the important concepts, and each document can be represented as a vector in this concept space.
- Selecting Sentences: Finally, LSA generates a summary by selecting the sentences that best capture the important concepts. This is typically done by comparing the similarity between each sentence vector and the concept vectors. The sentences that are most similar to the important concepts are selected to form the summary.

In [2]:
```
df = pd.read_csv('BIA_preprocessed.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	title	author	Date	text	P_text
0	0	Coinbase stock drops after SEC Wells notice, a...	Alex Wilhelm	March 22, 2023	American crypto giant Coinbase received a Well...	american crypto giant coinbase received well n...
1	1	A new Drake x The Weeknd track just blew up — ...	Amanda Silberling	April 17, 2023	A song featuring the voices of Drake and The W...	song featuring voice drake weeknd called heart...
2	2	SpaceX launches fully stacked Starship for the...	Darrell Etherington	April 20, 2023	SpaceX has launched its fully stacked version ...	spacex launched fully stacked version starship...
3	3	Children's data feared stolen in Fortra ransom...	Carly Page	March 28, 2023	The fallout from Fortra's mass ransomware atta...	fallout fortra mass ransomware attack continue...
4	4	Semgrep (formerly r2c) lands $53M investment t...	Ron Miller	April 18, 2023	Ideally, when it comes to building secure appl...	ideally come building secure application best ...

In [4]:
```
main_text = df['text']
summary_arr = []
```

In [5]:
```
for i in main_text:
    #first need to split article in sentences using . split
    temp_sen_arr = i.split('.')

    #convertion of sentences in TFIDF cevtors
    vectorizer = TfidfVectorizer(stop_words='english', use_idf=True, smooth_idf=True)
    X = vectorizer.fit_transform(temp_sen_arr)

    #Decomposition using SVD(support vector decomposition)
    U, S, Vt = svds(X, k=2)

    #creation of sentences to vectors
    sentence_vectors = np.matmul(X.toarray(), Vt.T)

    #select the number of important sentences for summary here we are summarizing with top most 3 important sentenc
    num_sentences = 3
    summary_indices = np.argsort(sentence_vectors.sum(axis=1))[::-1][:num_sentences]
    summary_indices.sort()
    summary = '. '.join([temp_sen_arr[i] for i in summary_indices])

    #append to summary array
    summary_arr.append(summary)
```

3.2) TextRank Algorithm:

TextRank is an unsupervised algorithm that ranks sentences in a document based on their importance. It works by representing the document as a graph, where the nodes are the sentences and the edges represent the similarity between the sentences. The algorithm then applies PageRank, a graph-based ranking algorithm, to the graph to determine the importance of each sentence. The top-ranked sentences are then used to generate the summary.

Step wise implementation of the algorithm:

- Sentence Extraction: The first step is to extract the individual sentences from the input document.
- Building a Sentence Graph: TextRank then builds a graph representation of the sentences in the document. Each sentence is represented as a node in the graph, and edges are added between sentences that are semantically similar. The similarity between two sentences is typically measured by the cosine similarity between their respective vector representations.
- Calculating Sentence Importance: Once the graph has been constructed, TextRank applies the PageRank algorithm to calculate the importance of each sentence. PageRank is a graph-based algorithm that was originally developed by Google to rank web pages in their search results. It works by assigning a score to each node in the graph based on the number and quality of the incoming edges. In the case of TextRank, the incoming edges represent the similarity between sentences, so the score of a sentence is proportional to the sum of the scores of the sentences that link to it.
- Selecting the Top Sentences: Finally, the top-ranked sentences are selected to form the summary. The number of sentences selected is typically a parameter that can be tuned depending on the desired length of the summary.

In [9]:

```python
def textrank_summary(text, num_sentences=3):

    #first need to split article in sentences using . split
    sentences = text.split('. ')

    #convertion of sentences in TFIDF cevtors
    vectorizer = TfidfVectorizer(stop_words='english', use_idf=True, smooth_idf=True)
    X = vectorizer.fit_transform(sentences)

    #creating cosine similarity matrix
    sim_matrix = cosine_similarity(X)

    #creating graph from the sim_matrix
    nx_graph = nx.from_numpy_array(sim_matrix)

    # Get the scores using the pagerank algorithm
    scores = nx.pagerank(nx_graph)

    # Sort the sentences by score and select the top sentences
    num_sentences = min(num_sentences, len(sentences))
    if num_sentences <= 0:
        return ''
    elif num_sentences == 1:
        return sentences[np.argmax(scores)]
    else:
        ranked_sentences = sorted(((scores[i],s) for i,s in enumerate(sentences)), reverse=True)
        summary = '. '.join([ranked_sentences[i][1] for i in range(num_sentences)])
        return summary
```

Analyzed Data:

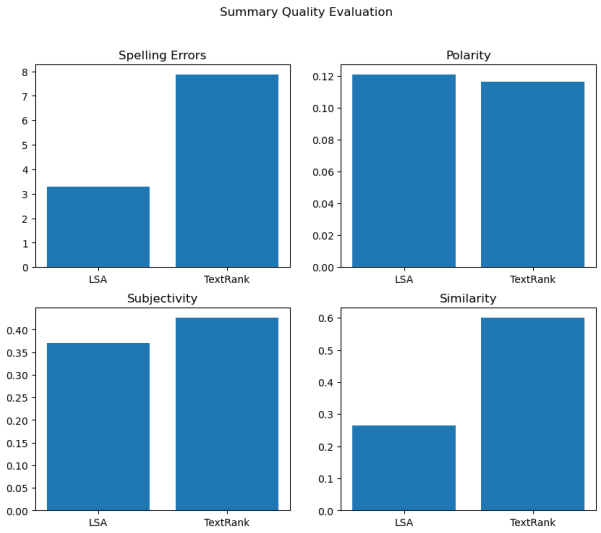
- The analysis of the bar chart comparing LSA and TextRank summarization algorithms showed differences in their performance.
- LSA had fewer spelling errors than TextRank in the same summarized article output.
- LSA had a higher polarity score than TextRank, indicating a stronger emotional tone.
- LSA had lower subjectivity compared to TextRank, indicating a more objective tone.
- TextRank had a higher similarity score than LSA, indicating more similarity between the original article and the summary.
- These observations highlight the differences in the summarization approaches taken by LSA and TextRank and provide insights into their respective strengths and weaknesses.
- In Conclusion the average of all scores like Spelling Errors, Polarity, Subjectivity and Similarity is compared and the best algorithm for Summarization is Used, In this case its TextRank

Spelling errors: Spelling errors refer to the incorrect spelling of words in a text document. They can occur due to various reasons such as typing mistakes, incorrect usage of homophones, and lack of knowledge of spelling rules. Identifying and correcting spelling errors is important to improve the readability and accuracy of a text document.

Polarity: Polarity refers to the sentiment or emotional tone of a text document. It can be positive, negative, or neutral. Polarity analysis involves using natural language processing techniques to analyze the sentiment of a text document. This analysis can be useful in various applications such as social media monitoring, customer feedback analysis, and brand monitoring.

Subjectivity: Subjectivity refers to the degree of personal opinion, bias, or interpretation in a text document. It can be used to classify a text document as objective or subjective. Objective texts contain factual information while subjective texts contain opinions, beliefs, and attitudes. Subjectivity analysis can be useful in various applications such as product reviews, news analysis, and opinion mining.

Similarity: Similarity refers to the degree of resemblance or similarity between two text documents. It can be measured using various techniques such as cosine similarity, Jaccard similarity, and Euclidean distance. Similarity analysis can be useful in various applications such as document clustering, duplicate detection, and plagiarism detection.




```
**********spelling Errors**********
Using LSA:-  3.3
Using TextRank:-  7.88
**********polarity**********
Using LSA:-  0.12112265254586684
Using TextRank:-  0.11618935380756809
**********subjectivity**********
Using LSA:-  0.3701167216381503
Using TextRank:-  0.42703113734149456
**********similarity**********
Using LSA:-  0.2641776294629183
Using TextRank:-  0.6012807037878003
```

Resultant Summarized News from Both Algorithm:

1. Actual Article:

'Lyft's newly appointed CEO David Risher told employees in an email Friday that the company is significantly reducing its workforce as part of a restructuring effort. Risher said the restructuring is part of Lyft's plan to "better meeting the needs of riders and drivers." The company confirmed that it has not changed its guidance for the first quarter in spite of the upcoming layoffs. What's less clear is how this might affect programs outside of ride-hailing such as its bike-sharing service. Lyft doesn't employ drivers who use the ride-hailing app to pick up and drop off riders. Instead, the layoffs will be directed at the company's more than 4,000 full-time employees. Employees will learn whether they have a job or not via an email that will be sent out April 27. Lyft wouldn't disclose the number of people who will be cut. A WSJ report, citing unnamed sources, said about 1,200 workers, 30% of its total workforce, would be affected. Risher, a former retail executive at Amazon, took over the CEO position at Lyft after co-founders Logan Green and John Zimmer stepped down last month. Risher explained in the email that he made the decision to help the company achieve its two core purposes. "Lyft has two purposes that are linked to each other: We help riders get out and about so they can live their lives together, and we provide drivers a way to work that gives them control over their time and money," he wrote. The move may come as no surprise to those who closely follow Lyft and its struggles to keep apace of rival Uber. Risher told TechCrunch in a late March interview that Lyft might drop its shared rides offering and make other changes to its business model in a bid to focus on its core ride-hailing business and become profitable. He listed a number of other products and services that could disappear, including Wait & Save, which allows riders in certain regions to pay a lower fare if they wait for the best-located driver. "It's possible that maybe we don't need both of those anymore and that we can focus all our resources on doing a fewer number of things better," Risher told TechCrunch at the time. "Maybe it's time for us to say the shared rides were great for a time, but it's time to let that go.'

2. Latent Semantic Analysis (LSA) Output:

' What's less clear is how this might affect programs outside of ride-hailing such as its bike-sharing service. Lyft doesn't employ drivers who use the ride-hailing app to pick up and drop off riders. He listed a number of other products and services that could disappear, including Wait & Save, which allows riders in certain regions to pay a lower fare if they wait for the best-located driver'

3. TextRank Output:

'Lyft's newly appointed CEO David Risher told employees in an email Friday that the company is significantly reducing its workforce as part of a restructuring effort. Risher said the restructuring is part of Lyft's plan to "better meeting the needs of riders and drivers." The company confirmed that it has not changed its guidance for the first quarter in spite of the upcoming layoffs. Risher told TechCrunch in a late March interview that Lyft might drop its shared rides offering and make other changes to its business model in a bid to focus on its core ride-hailing business and become profitable'

	Unnamed: 0.1	Unnamed: 0	title	author	Date	text	P_text	summary_using_LSA	summary_using_textrank
0	0	0	Lyft to make 'significant' cuts across ride-ha...	['@kirstenkorosec']	April 21, 2023	Lyft's newly appointed CEO David Risher told e...	lyft newly appointed ceo david risher told emp...	What's less clear is how this might affect pr...	Lyft's newly appointed CEO David Risher told e...
1	1	1	Roku gains 1.6 million active streaming accoun...	['@aiishamalik1']	April 26, 2023	Roku delivered its first-quarter results on We...	roku delivered first quarter result wednesday ...	6 million. Accordingly, we expect the adverti...	Notably, the company revealed that it reached ...
2	2	2	Lookout sells its consumer cybersecurity busin...	['@psawers']	April 26, 2023	Lookout's long-running transition to becoming ...	lookout long running transition becoming enter...	" For F-Secure, the deal gives it a stronger f...	Lookout's long-running transition to becoming ...
3	3	3	Bend is taking on Brex and Ramp with a green t...	['Tim De Chant', 'Alex Wilhelm']	May 3, 2023	When the SEC announced that it planned to requ...	sec announced planned require company purview ...	That lineup may change, of course. Brex, Ram...	Carbon credits are transacted through Patch, t...
4	4	4	TechCrunch+ roundup: 3 key hiring metrics, bui...	[]	March 31, 2023	The expense involved in recruiting, training a...	expense involved recruiting training onboardin...	Because it takes "about 15 touches for a pros...	Thanks for reading, Walter Thompson\nEditorial...

Discussion:

1) What motivated the development of QuickRead?
Answer: The overwhelming amount of news available on the internet and the difficulty of finding time to read lengthy news articles motivated the development of QuickRead.

2) How does QuickRead work?
Answer: QuickRead uses natural language processing techniques to extract the key information from news articles and generate a summary that captures the essence of the article.

3) What are some potential benefits of QuickRead?
Answer: QuickRead makes it easier for users to consume news quickly and efficiently, saving time and allowing individuals to stay informed about current events more easily. It may also benefit the news industry by increasing readership and engagement.

4) What are some potential drawbacks of QuickRead?
Answer: QuickRead summaries may not capture the full context or nuance of a news article, leading to incomplete or biased understanding of the content. Additionally, relying solely on summaries may lead to a lack of in-depth knowledge on certain topics.

5) How accurate and reliable are QuickRead summaries?
Answer: The accuracy and reliability of QuickRead summaries depend on the natural language processing techniques used to extract information and generate the summary. It is important to continually evaluate and improve the accuracy of the technology to ensure the summaries are as reliable and accurate as possible.

6) What is the impact of QuickRead on the news industry?
Answer: QuickRead has the potential to increase readership and engagement with news articles, as well as provide a new revenue stream for news organizations. It may also encourage individuals to stay informed about current events more easily.

7) How might QuickRead be improved or expanded in the future?
Answer: QuickRead could be improved by incorporating user feedback and continually refining the natural language processing techniques used to generate summaries. It could also be expanded to summarize other types of content, such as academic papers or research articles.

Future Works:

- Incorporating user feedback to improve the accuracy and effectiveness of the summarization techniques used in QuickRead.
- Integrating additional features, such as sentiment analysis, to provide users with a more comprehensive understanding of the news articles they are reading.
- Exploring the use of multi-document summarization techniques to provide users with a more complete view of a particular topic or event.
- Investigating the use of machine learning algorithms to automatically identify the most important information in a news article and generate a summary that accurately captures its essence.
- Developing a mobile application or web-based platform that allows users to access QuickRead summaries on-the-go, and explore additional features such as customizable filters, topic clustering, and more.