Analysis of News Repetition in Daily News Bulletin

Submitted By:
Tanmay Kumar Shrivastava (12241870)
Uday Bhardwaj (12241910)
Shivam (12241810)

Semester: 2023-24 M

Index

Introduction to the Problem Statement

From Earlier days, Information has been one of the most powerful tools for Humans. The importance of the information can be reflected by its usefulness. Earlier our food and survival depended on the gathering of information. Even today, it is equally important.

In the earlier days, we resorted only on the information which was collected by the individual or by known people. However, over time, The methods of gathering the information have changed vastly. In Today’s era, News channels are among the primary sources of getting general information from around the World. One of the most popular variants of the Modern news coverage is the Daily news bulletins which provide information about the latest happenings for that particular day.

Due to a drastic increase in the online news sites and applications, News channels have seen a decrease in their audience over time(1). Despite such occurrences, News Channels still are a primary source for news.

Now arises the problem of the ***Repetition of News*** in the news reports. This is one of the two major problems, the other one being the validity of the news reports being shown. However, In this report We focus on the problem of ***News Repetition*** in the Daily news Bulletins of the News Channels.

Abstract

The project aimed at finding out the similarities and differences between the non-stop news headlines. However, this easy-looking task proved to be the biggest hurdle faced during our project. There were ample datasets related to news but none of them provided a recent and collection of articles. This led us to narrow down our target. We still managed to get the data in a format which would be suitable for the task of analysis and visualization.

After the data pre-processing, we moved forward to the analysis of the available data. This was done with the help of pre-trained models and rich libraries of python. During some of the analytical parts, it was computationally expensive. However, we tried to reduce the commutation time through various methods.

Finally, we got the results from the data and we used the various visualization libraries of Python to visualize the different results.

Overall, After facing various challenges, we finally got the results, although limited, which the problem statement demanded.

Workflow

1. Data Collection
    a. Approach used
    b. Final Dataset
2. Data Pre-Processing
    a. Reducing Computation time
    b. Lemmatization
3. Analysis
    a. General Description
    b. Similarity techniques
4. Visualization

Data Collection

When we proceed to any data-related task, the sub-task of Data collection comes out to be the most tedious task in the project. However, this could be reduced by using previously available datasets from the web.

Considering our problem, we required a dataset which contains the attributes such as the name of the channel, news bulletins for a particular date in separate rows,

Initial Approach

Our Initial approach was to find a previously available dataset for the task. During our search, we encountered few datasets related to the topic but there were some limitations which were either in the dataset or in the method to obtain them. We will just briefly discuss one of them as follow:

- We found a dataset on Kaggle which contained news articles published by a News Channel during a time-span of 10 years. However, the dataset contained the news articles which were already distinct and had very less similarity in them.

Moving on, we tried to obtain the data from the Youtube videos of Indian News Channels directly. For this task, We required a model which could either scrap the subtitles of the videos or could convert the voice to text from the video.

Scrapping Subtitles

For this task, we used the youtube-transcript-api authored by Jonas Depoix(2). We used this API with the help of the Python programming language.

Working of the API: The API does not have the requirement for a headless browser like most other chromium-based solutions have. It can be directly used via cli but we have focused on using the API with Python due to the rich analytical libraries offered by Python.

The API gives a dictionary which consists of the time intervals of the subtitles and the subtitles themselves. This was then used to get a complete set of news headlines through the help of different data structures of Python.
.

Difficulties encountered: The API does not provide support for videos which do not have their subtitles on and can only transcribe the videos for the Channels which allow it.. This was one of the major difficulties faced as Most of the Indian News Channels do not allow transcripting of their news bulletins.

Video to text transcribers

Owing to the non-working of the API on the Indian News Channels, we proceeded to the usage of the web transcribing services which were available.

Difficulties encountered: There were a few difficulties faced which are written as follows:

1. *Limited Free Usage:* Nearly every web-based transcribing application is limited for free usage. They offer a monthly or yearly subscription plan which is expensive for normal students.This can be considered due to the computation required for the task of transcription. We tried to utilize the limited usage time of the applications. This leads us to our second difficulty.
2. *Inaccuracy:* There was a lot of inaccuracy encountered when using the free and limited versions of the above mentioned applications. For reference, In one such case the returned text was itself repeating a certain news headline for half the article.

Thus, we decided to leave this approach because of the transcribed data obtained which could produce a major error in our findings.

Before proceeding on to the answer for our search for a Dataset, another problem we faced was that after getting the initial 100 news manually, we tried to classify the different headlines into different clusters. The outcome of this method was that the clustered group themselves contained the similar news headlines which would render our main objective of getting a similarity percentage among the headlines.

Now we discuss about the data collected at last and the method used.

Breakthrough

During our search for the news Bulletin videos from which we could scrap the headlines, we came across the European news channel Euro News. After confirming that the news channel provided the transcription through the API, our next sub-task focused on finding suitable videos which could be used as the Standard data and could be compared easily among each other

This problem was solved by observing that the news channel gave different news bulletins for the 3 different timings of the day namely *Morning, Midday* and *Evening.* Then, we collected such data for about 512 days over a time span of 2 years removing any day which was missing any news headline for the day.

Final Dataset

The dataset made after collecting the data has the following attributes:-

INDEX	NAME	DETAILS
1.	Channel	Channel from which news has been transcribed
2.	Date	The date on which the Headlines broadcasted
3.	Time	The timing of the Broadcast- Morning, Midday, Evening
4.	Location	Based country of news channel: India/ Outside India
5.	Headlines	Raw headlines collected

Total number of rows: 1536

Each set of news headlines in every row was in English. In the next section, we will discuss the cleaning done in detail on the above dataframe.

Data Pre-processing

Data preprocessing is a part of data preparation which can be described as some type of processing/operations performed on raw data to make it less computably expensive or more undestandable. It is a very important step before proceeding to the task of Data Analysis. Recently, data preprocessing techniques have been adapted for training machine learning models and AI models and for running inferences against them(3).

In our task of Data pre-processing, we limit ourselves to the traditional data Pre-processing techniques. We have used the below given libraries of Python for this task.

- Pandas
- nltk (Natural Language toolkit)
- re (Regular Expressions)

Reducing Computation time

In the dataset obtained from the collection, we had raw news headlines without any operation done on them. Each set of Headlines consisted of many lines along with the noise which is described later and operating any analysis task on them would prove to be very expensive from the view of Computation.

Below, we have described the task and the procedure which we used to solve it.

1. **Task:** Each set of headlines contains words which are common in English i.e. verbs, adverbs and adjectives. These were the words which do not add any new information to the overall meaning of the headlines.

   Solution: Solving the above problem required the removal of such words from each set of headlines.

   We used the stopwords set from the nltk library and removed any stopwords which were present in the headlines. This proved to be effective as we could reduce the size of the headlines which will be usd later for analysis.

2. **Task:** We found that there was a lot of noise in the headlines, largely due to words which were added by the API. After observing these kinds of noises, we figured out that these noises have different patterns, for example, In most of the headlines, [MUSIC] was added due to the time-frames recorded by the API when it extracted subtitles.

   Solution: All such types of Noises could be easily removed with the help of Regular Expressions. Hence, we used “re”- the regular expression library of Python to overcome this limitation.

By the above mentioned solutions, we could reduce the computation time of our code very less compared to if we operated it on the raw data of Headlines.

Lemmatization

Lemmatization is the process of considering the different forms of a word together so they can be analyzed as a single word. It is similar to Stemming(4). Many other operations such as removing unnecessary suffixes from words also come under Lemmatization and Stemming.

Although the accuracy of the API used for scrapping the subtitles is high, there were some words which were repeated in the original subtitles itself. This could be reduced by using Lemmatization.

Preference over Stemming
We have preferred Lemmatization over Stemming as the latter one affects the meaning of the word up to a certain limit. Lemmatization can be considered a moderate version of Stemming.

Data Analysis

General Description

Now, we elaborate about the main part of our project. The Analysis techniques shown in this section are the base on which we will classify our observations and visualize them later.

For the sub-tasks in this section, we have used the *Sentence-transformers* library of Python which has models with rich features which can be used for different tasks such as embeddings and Similarity detection.

Objective
In this section, our main objective was to get certain credentials which could help us in pointing out the similarities and dissimilarities in the news headlines for a particular day.

Approach
To fulfill our objective, it would be easier if we could label some similarity value to our data comparisons which would make it easy to compare and use for the different Visualization techniques later.

Usage of Data
The dataset which we obtained from the earlier section consisted of rows which have different features previously mentioned. Our major focus here is on the following features:

- Date
- Timing
- News Headlines

Procedure
Here we give a brief and an informal algorithm which we have used in completing our objective.

1. First, we decided on comparing among the different timings for a particular day. For example, Let us say there were 3 news bulletins for 1 January 2022, we compared between the distinct pairs which will be
    - Morning-Midday
    - Morning-Evening

- Midday-Evening

2. After deciding the data which we need to compare, we focused on the data structure which we will use. Considering we required our data in a manner which can be later converted into a dataframe, we implemented Dictionary. Next, I provide a brief description of the dictionary- Keys and Values. Before proceeding, we discuss the values. The values for initial keys are lists themselves. These lists store the similarity percentages, dates, technique used and the timings compared.

Keys	**Values**
Date	A list which stores the date for which the similarity scores are calculated.
Technique	A list which shows the technique used
Morning-Midday	A list comprising the similarity value for the headlines from Morning and Midday news bulletins
Midday-Evening	A list comprising the similarity value for the headlines from Morning and Midday news bulletins
Morning-Evening	A list comprising the similarity value for the headlines from Morning and Midday news bulletins

3. We grouped the data frame by a particular date and obtained a sub-data frame corresponding to the three different timings of the day. A Point to note here is that we have already ensured that our data frame has 3 different timings for every date present in the date frame. Otherwise, it would have later led to erroneous code.
4. Finally, we obtained the similarity values from two different similarity techniques.

Similarity Techniques

Before giving a description of the similarity techniques which we have used, It remains important to discuss the models which we have used to convert our set of Headlines into a format which can be used as an input to these similarity techniques. We do this by first explaining about the format, known as Embeddings and, detail the two models which are used to convert the set of Headlines into Embeddings.

Embeddings

In simple terms, Embeddings can be referred to as a lower-dimensional space or set in some cases to which we can convert any higher dimensional data such as text, video. For our case, we have used Numerical Word Embeddings. Numerical Word Embeddings are a lower-dimensional space where individual words are represented as real-valued vectors in a lower-dimensional space. Each word is represented by a real-valued vector with tens or hundreds of dimensions. These types of embeddings have become an important part of modern data analysis for text documents.. It maps the unstructured text data onto lower-dimensional numerical vectors while retaining key characteristics of the original unstructured text. After that, we can use these embeddings to compare between the text for their similarity.

Models

We have used two different models for the different kinds of similarity techniques

1. Cosine Similarity: `all-MiniLM-L6-v2`
2. Semantic Similarity: `clips/mfaq`

Both of the above models are taken from Hugging face and are used with the help of the *Sentence-transformers* library of Python.

A Brief Description of Models

The models we have used are to provide us with embeddings. They are pre-trained models which are only used in the project. The selection of the particular models for applying with a particular similarity technique is chosen with respect to similar previous tasks of texts similarity detection. Depending on the source of the text, the accuracy of the model for a particular kind of similarity can vary slightly as we shall see in the case of Semantic similarity.

- *all-MiniLM-L6-v2*
  It maps sentences & paragraphs to a 384 dimensional dense vector space and can be used for tasks like Cosine similarity or semantic search. The model is intended to be used as a sentence and short paragraph encoder. Given an input text, it outputs a vector which captures the key information. The sentence vector may be used for information retrieval, clustering or sentence similarity tasks (7).

  During the use of the model, we passed the news headline as the input and it would create a set of low-dimensional vectors which could be further used for the comparison between such vectors to obtain the similarity score. We applied this model for finding the Cosine Similarity score.

One advantage this model provides is that it can be used to embed sentences which have large length. This was highly important as the set of News Headlines obtained still remained large.

- *clips/mfaq*
  It also provides embeddings. The model is trained from the available FAQ dataset. It produces a set of vectors which can be further used to compare between each other to find the similarity scores (8).

  Similar to the previous model, we have used the model to provide us with the embeddings of a particular news headline which would be used to compare against each other to find the similarity. We have used this model for the calculation of the values for Semantic similarity.

Methods for finding Similarity

For the comparison of the news headlines, we have used two similarity techniques-
- *Cosine Similarity*
- *Semantic Similarity*

Cosine Similarity

Generally, Cosine similarity is used to measure the similarity of two vectors. Specifically, it measures the similarity in the direction of the vectors ignoring differences in their magnitude/scale. The similarity of two vectors is measured by the cosine of the angle between them (5).
The cosine similarity can be calculated from the method which is provided by the util class. It takes the input of two embedding vectors and gives out a 2-dimensional list with a single tensor element. The value in the tensor element is a decimal value less than 1 and greater than 0, at least for our comparisons. It is an indication about the similarity between the sentences for which we have calculated the embeddings.

<u>*Extraction and Normalization of values from tensor*</u>
The Extraction of the values can be easily done by dereferencing the single element present in the two-dimensional list. Further, the value inside the tensor can be easily obtained by using the *item* method of the tensor.

Next, we normalized the values by multiplying them by 100 which can be interpreted as the percentage of information which was similar in the two sentences.

Semantic Similarity

First we will explain about Semantic search. Semantic search tries to improve search accuracy by understanding the content of the search query. In contrast to traditional search engines which only find documents based on lexical matching, semantic search can also find synonyms.

In our sub-task, Instead of using the semantic search as a query and answer's accuracy problem, we have used it to compare between the two sets of embeddings which are provided by the model.

When we did the analysis by the use of Semantic search, we observed that the similarity values were very high compared to the similarity values obtained from the Cosine similarity. This can be due to the fact that similar news with new additions are reported about a certain incident that happened on a particular date.

Here, we used the *semantic search* method provided by the *util* module of the sentence-transformers library in Python. It required at least two inputs: one which will be a query and another which will answer. The method returns a dictionary containing the list of answers. Each sub-dictionary has "Score" as a key which contains a tensor with a value between 0 and 1. This would imply the similarity between the query and the answer.

<u>Extraction and Normalization of values from tensor</u>
We extracted the values directly from the dictionary and normalized it to give a percentage of similarity between the two news Headlines.

Data Visualization

For visualization, we have used Matplotlib and the Seaborn library. For greater understanding of results obtained by analyzing the dataset, we have used different plots for visual representation of results.

Following types of Data visualization plots are used:

1. ***Heat Map***
   It is a Data Visualization technique that represents the magnitude of individual value within the dataframe as a color.it is made for showing correlation between two features.

2. ***Scatter Plot***
   It is used to observe the relationship between two features.We use dot for representing the features value.It represents the value of another variable corresponding to the value of one feature.

3. ***Word Cloud***
   Technique is used to know the frequency of words in a sentence then we use Word Cloud.Word size is directly related to word frequency.

4. ***Bar Graph***
   Bar-Graph is a pictorial representation of Data-Frame with rectangular bars with heights proportional to the values that they represent.

5. ***Pie Chart***
   A pie chart is circular representation

We have the following Data Frame which contains Columns-

`'S.No','Date', 'Technique', 'Morning-Midday','Midday-Evening', 'Morning-Evening', 'Average Similarity'`

Data Frame

S.No	Date	Technique	Morning-Midday	Midday-Evening	Morning-Evening	Average Similarity
1	2023-01-01	Cosine Similarity	73.351413	44.613236	34.173316	50.712655
2	2023-01-01	Semantic Similarity	96.062630	97.445554	95.561874	96.356686

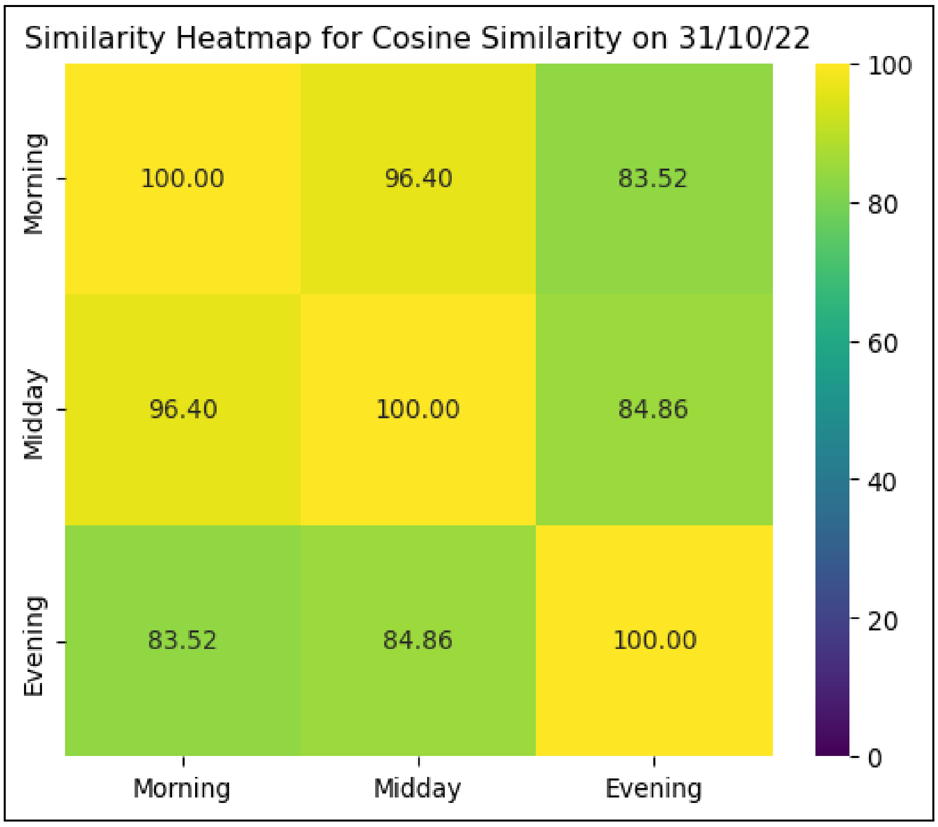
3	2023-01-02	Cosine Similarity	57.157195	59.162945	42.400160	52.906767
4	2023-01-02	Semantic Similarity	98.220742	98.225093	98.085129	98.176988

`'Morning-Midday'`-Similarity between news bulletin of Morning and Midday
`'Midday-Evening'`-Similarity between news bulletin of Midday and Evening
`'Morning-Evening'`-Similarity between news bulletin of Morning and Midday

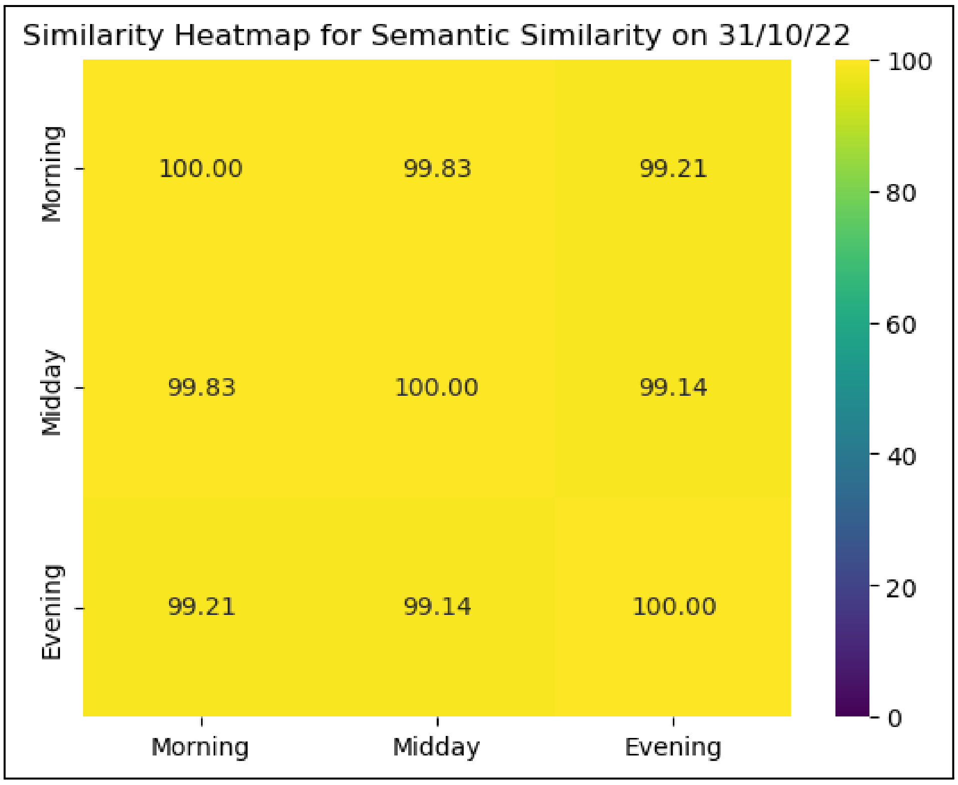
A. First to see similarity of a particular day we use heat map :

INPUT- DATE AND TECHNIQUE

Example: *Date-31/10/22 Technique-Cosine Similarity 31/10/22*



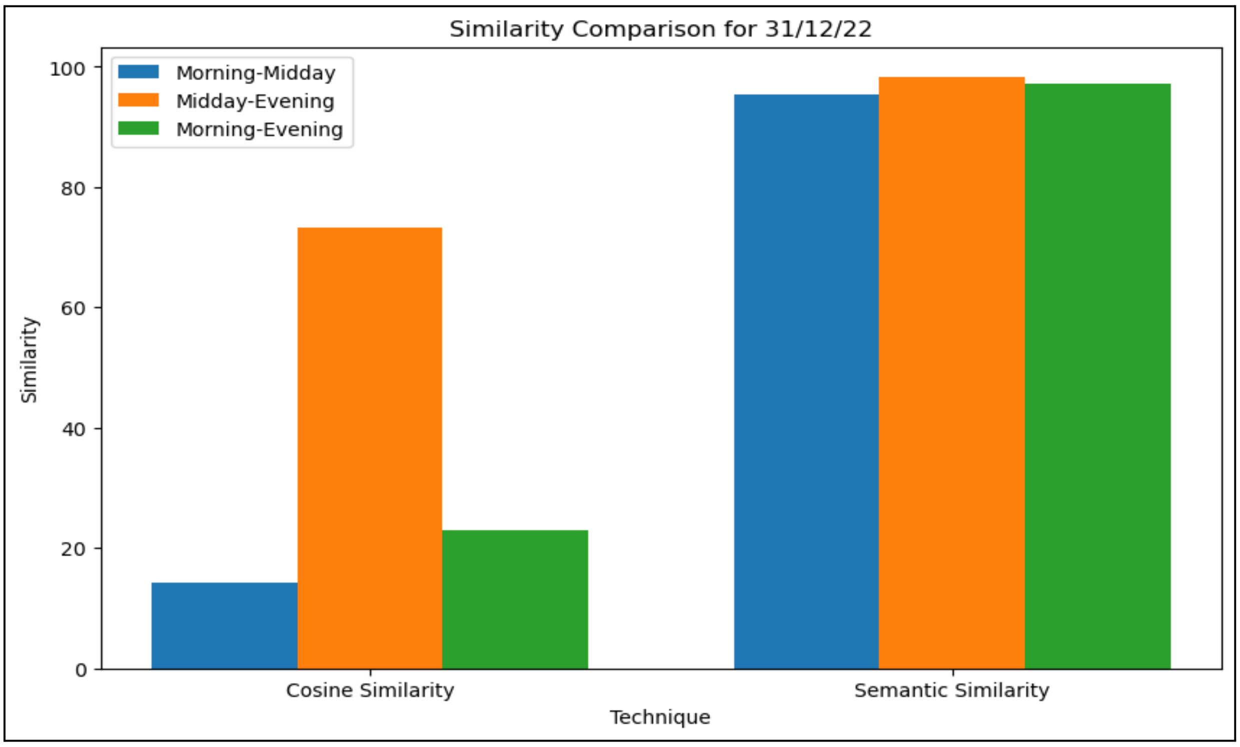
Example: Date-31/10/22 Technique-Semantic Similarity 31/10/22



B. To compare the similarity of a particular day we use Bar Chart

INPUT-DATE

Example : Date-31/10/22



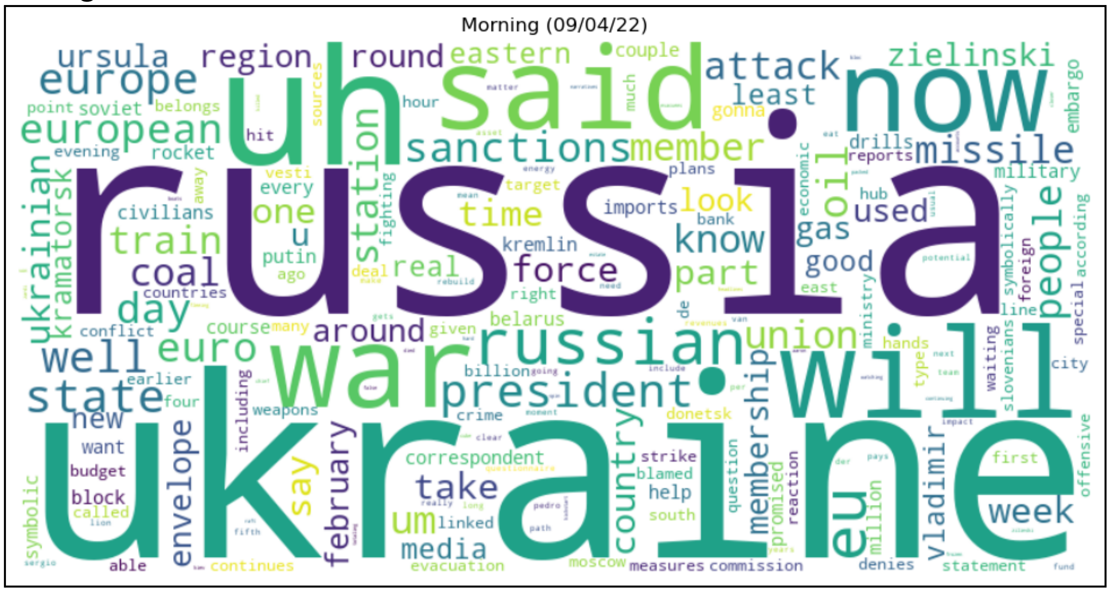
C. To See the particular day most frequent word used we use Word Cloud :

INPUT - DATE

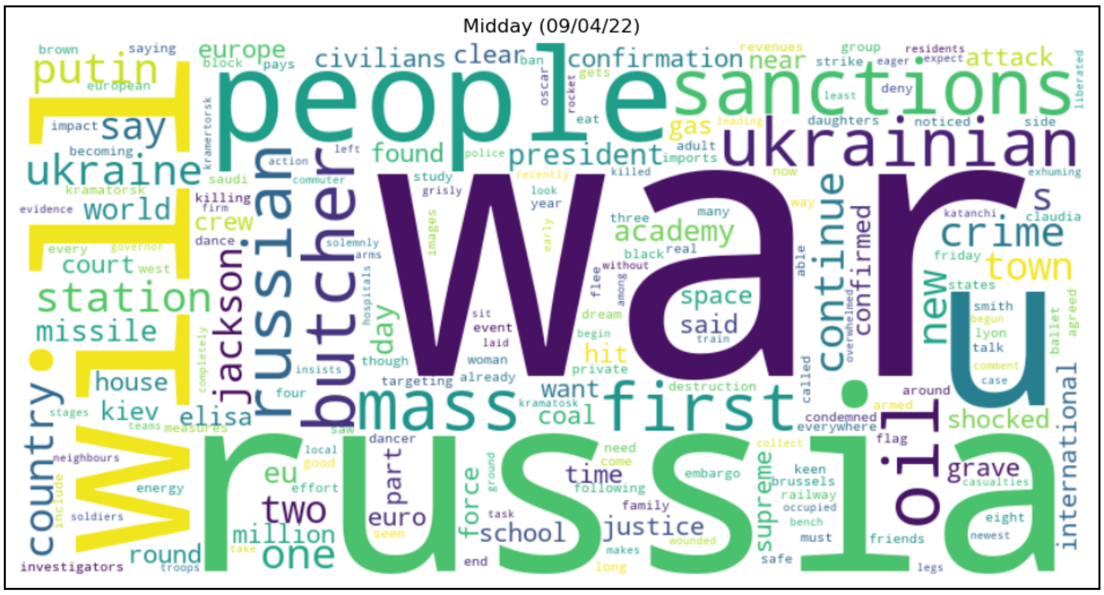
Example : 09/04/22

We see that in April 2022 Russia-Ukraine War is going-on . So the frequency of words like Russia ,Ukraine ,war ,kill are very high. Hence, Similarity between Morning, Midday, Evening is :

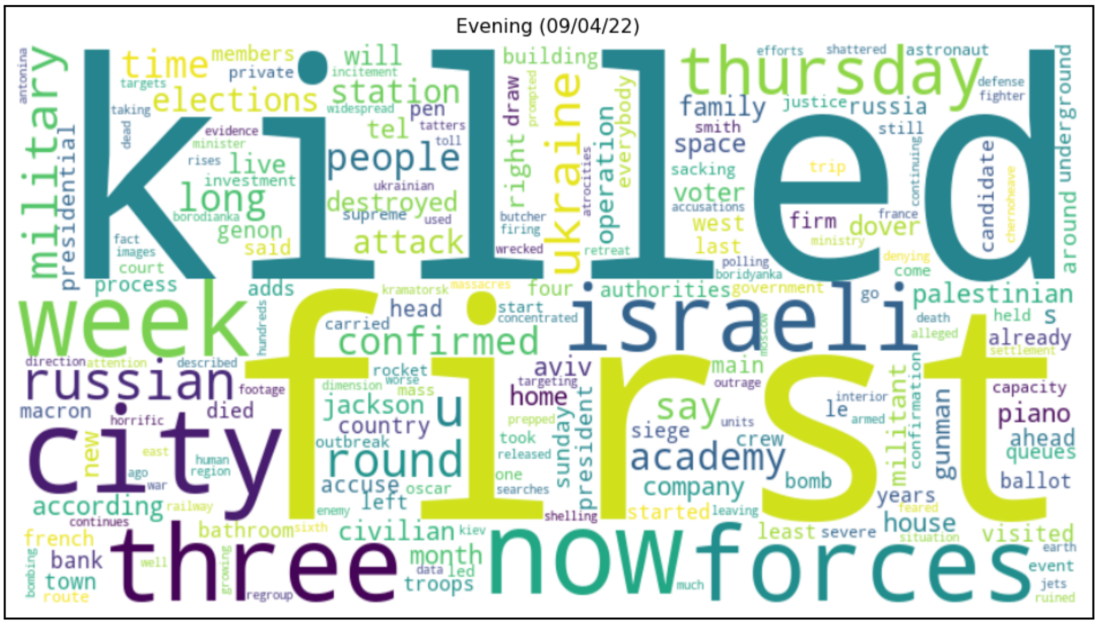
1. Morning



2. Midday



3. Evening



Observations-

1. The word clouds is dominated by words like ‘russia’, ‘ukraine’, ‘war’ etc. This means that the word cloud in the morning, midday, evening news is quite dominated by the topics of Russia-Ukraine war and many other topics.
2. The word clouds is a mixture of positive and negative words.
3. Similarly, in the evening word cloud, topics of Israeli-Gaza war/conflict can be seen.

09/04/22	Cosine Similarity	Semantic Similarity
Morning-Midday	71.69	97.77
Midday-Evening	70.69	98.40
Evening-Morning	68.61	98.34

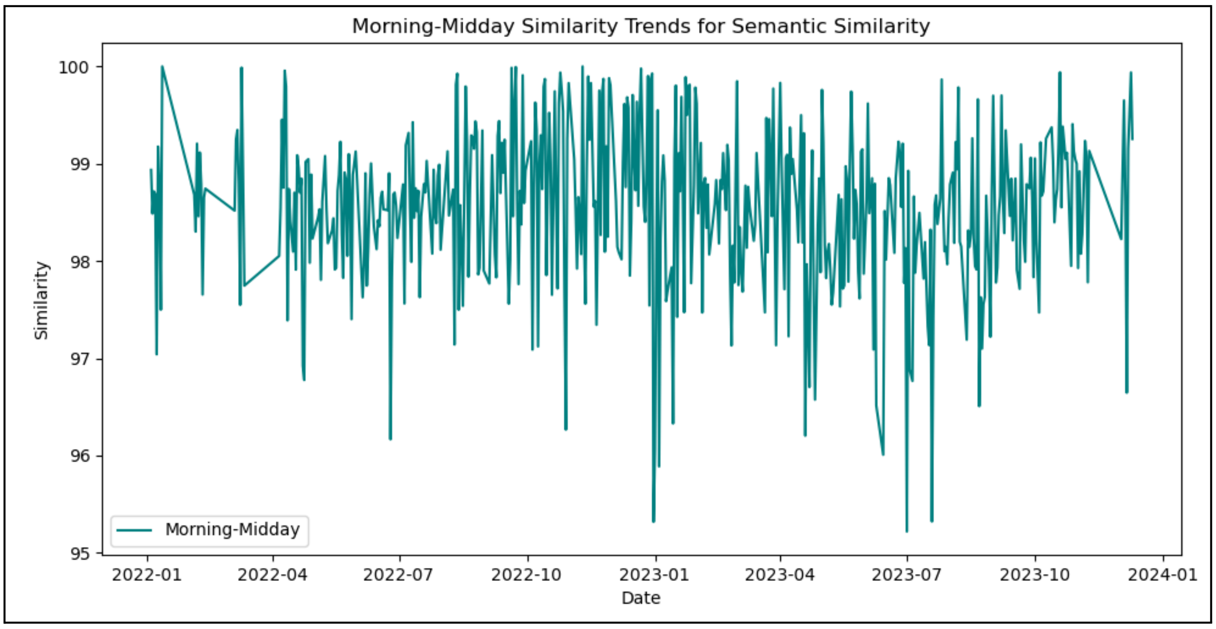
Here, we see that there are huge differences in values for both of the similarities.

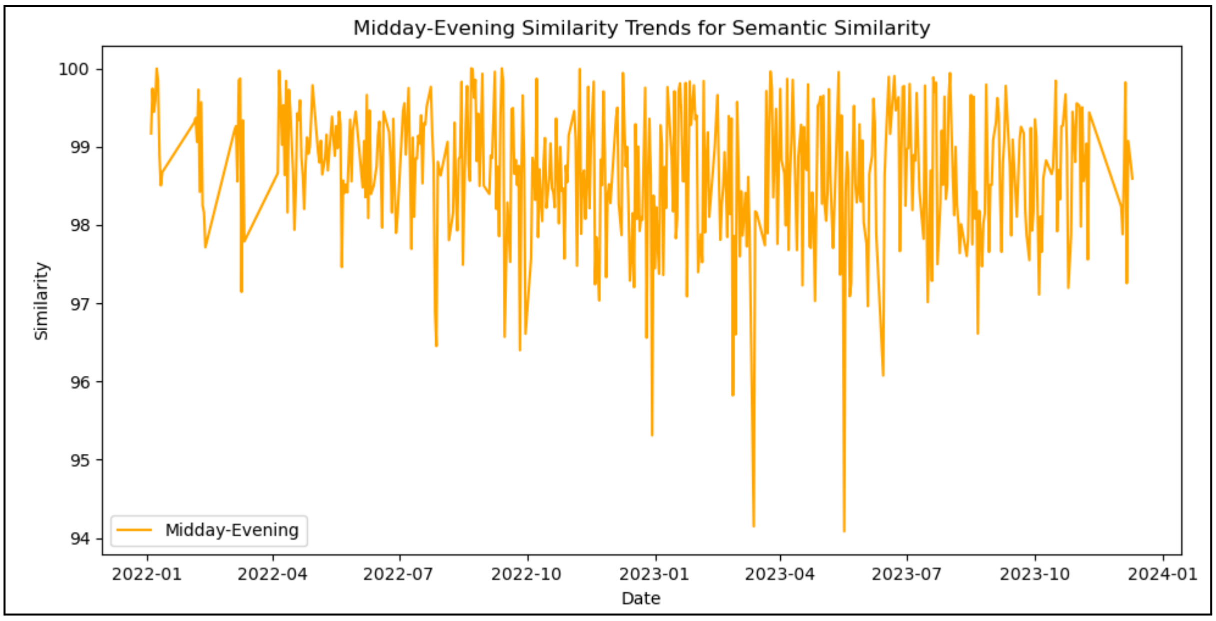
1. The cosine similarity is quite sensitive to the order of the words and the way the text has been written, while semantic similarity is not.
2. Despite the differences in the concepts of the similarities, the semantic similarity requires the meaning of the words/values extracted, while cosine similarity does not. Also, semantic similarity won’t be able to analyze some similar or unfamiliar words.
3. Semantic similarity is difficult to compute whereas cosine similarity is simple.

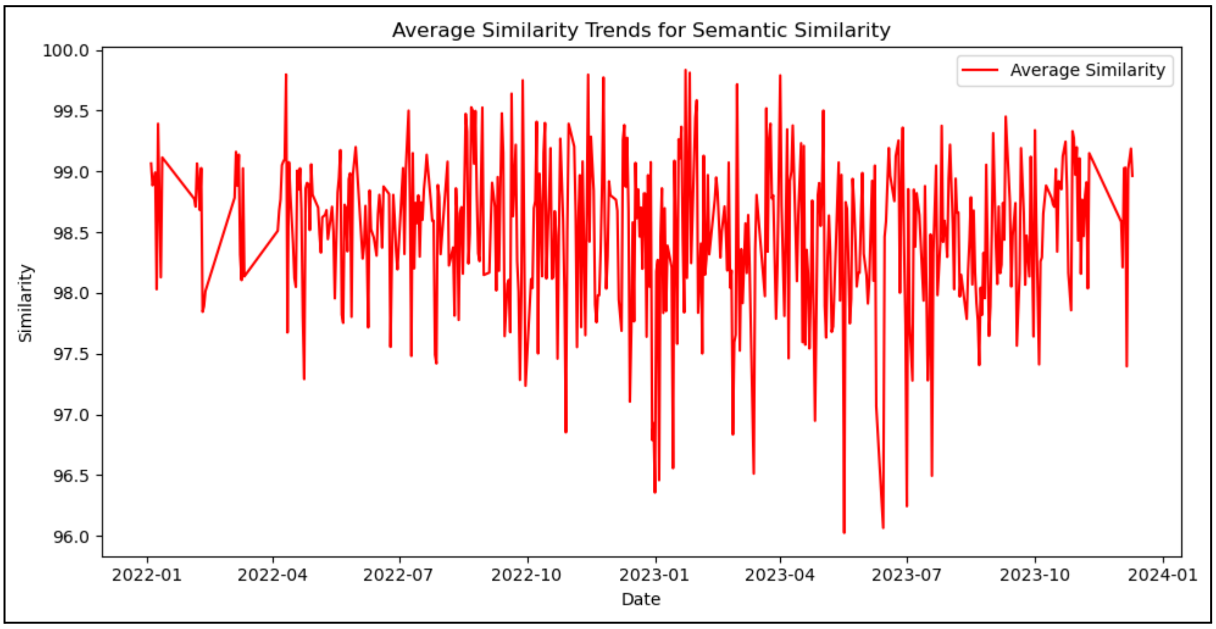
Overall, Semantic similarity is quite an accurate way to describe the similarity but We get more accurate results by applying cosine similarity. Another reason for the difference in the similarities can be attributed to the information addition in the latest news.

For example, let’s say there was a big accident in the morning. The channel reported the news of the accident right away in the morning news bulletin. By afternoon. There was an addition of information in the news related to the accident. Then, the channel would report the additional information along with the original news headline as a reference. Cosine similarity will give a lesser similarity value than the Semantic similarity value as Cosine similarity will find the distance between the vector embeddings whereas semantic similarity will give the distance according to the overall sense of the words.

D. To See the plot: how midday and morning similarity vary over period of time we use Time Series Plot:







It represents variation of similarity over a period of time.

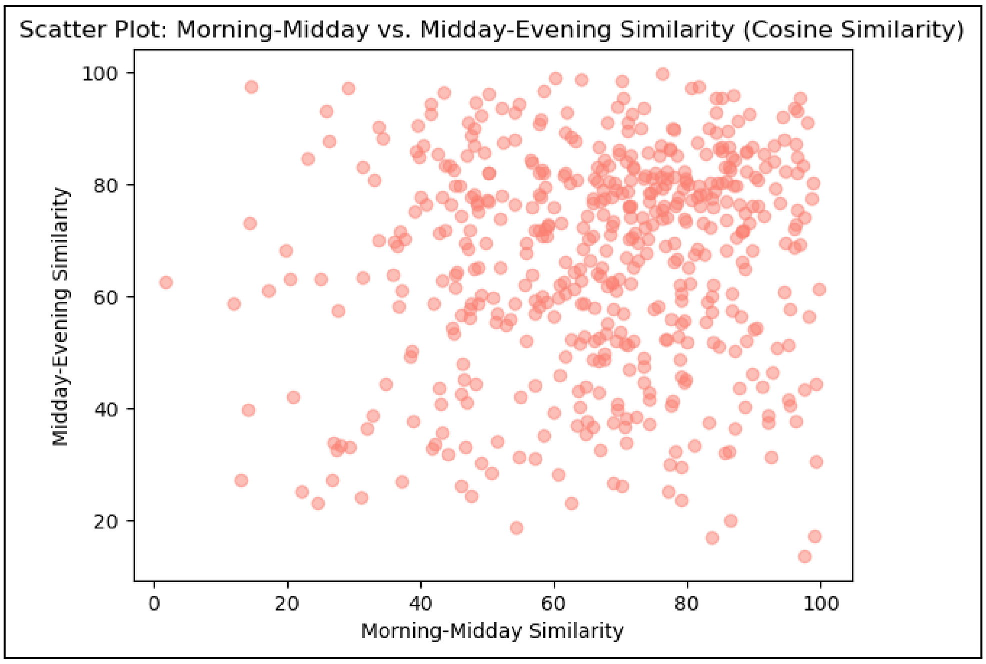
Cosine Similarity

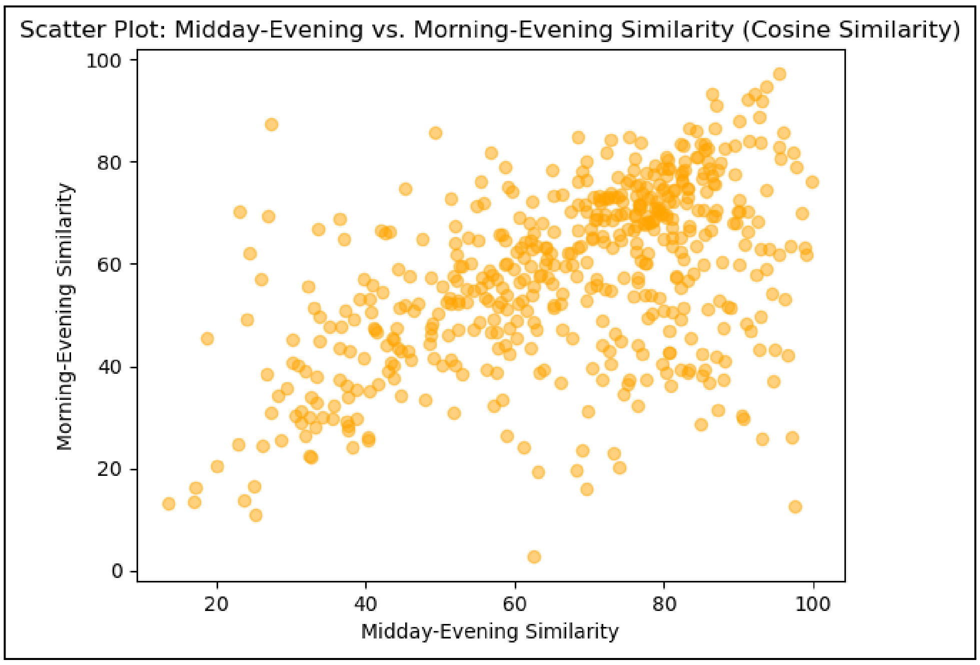
	Morning-Midday	**Midday-Evening**	**Evening-Morning**
Max	99.68	99.86	97.29
Min	1.80	13.57	2.76
Mean	66.75	66.47	57.51

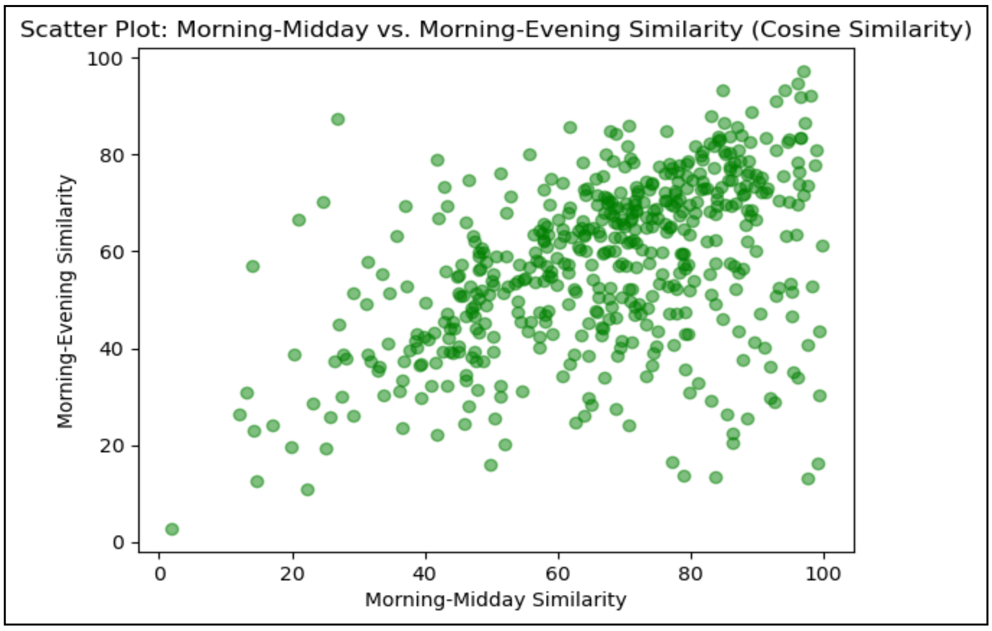
Semantic Similarity

	Morning-Midday	Midday-Evening	Evening-Morning
Max	100	100	98.8
Min	95.21	94.08	94.54
Mean	98.55	98.66	98.25

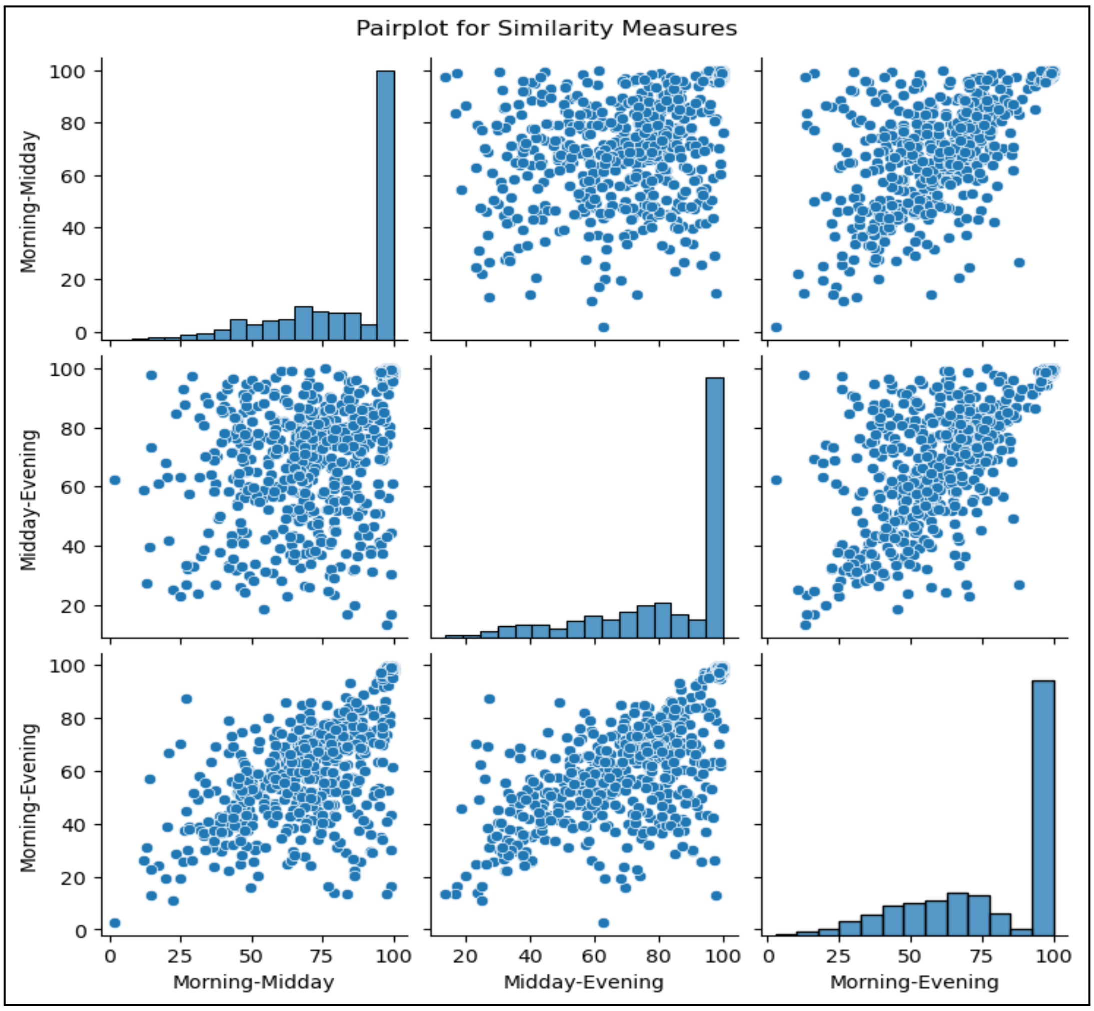
E. To See the plot how midday and morning similarity vary with each other we use Scatter Plot :



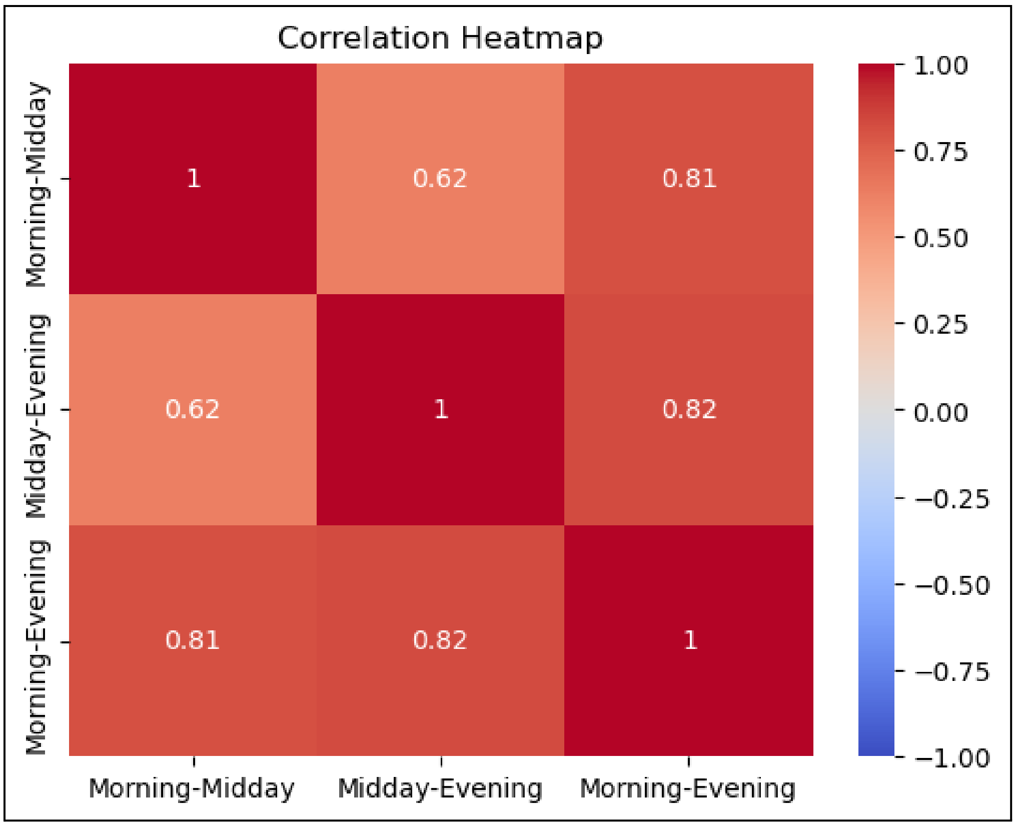




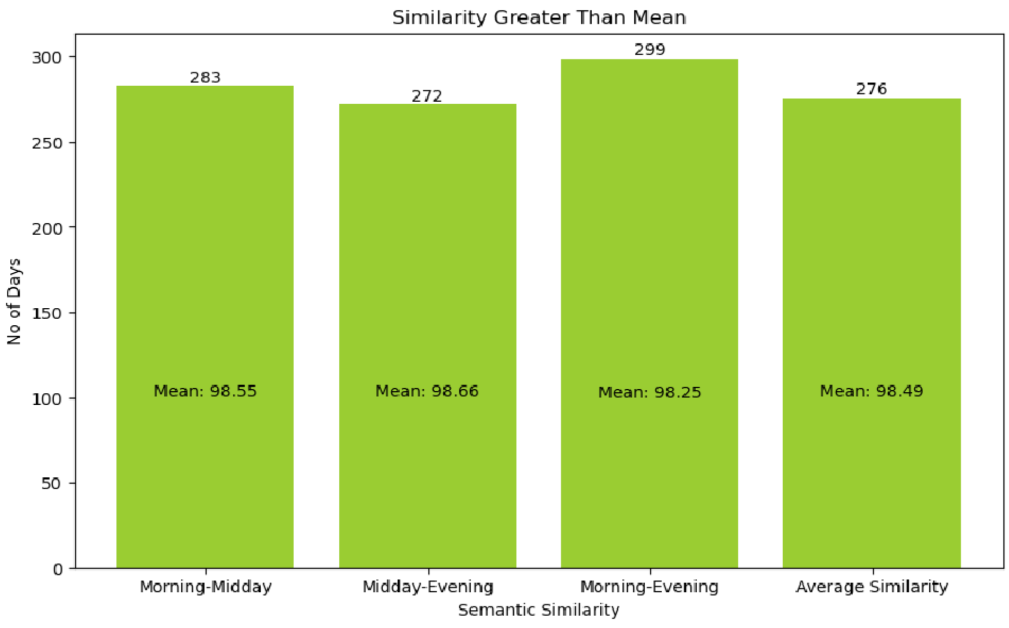
Scatter Plot is used to observe the relationship between two features. We can see that most of the points vary linearly.

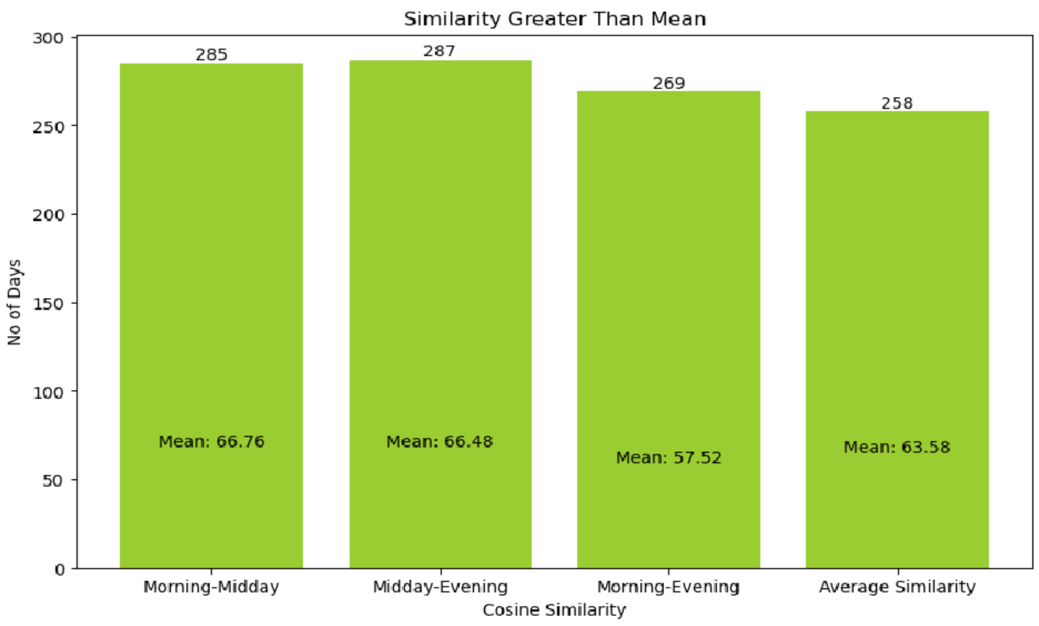


We can also Observe The Relationship between two variables by a pair plot.



F. Values Greater than Mean




Similarity Greater Than Mean
No of Days
300
250
200
150
100
50
0
285
287
269
258
Mean: 66.76
Mean: 66.48
Mean: 57.52
Mean: 63.58
Morning-Midday
Midday-Evening
Morning-Evening
Average Similarity
Cosine Similarity

Conclusion

From this project, we have learned that if the news is occurring in the whole world, the similarities may increase potentially. This is because of the following-

- Words are mostly similar, suggesting that there is a lot of similar coverage in the news channels.

- People all over the world are sharing the news, so the chances of people from different countries encounter the same information.

- Media is globalized, every news channel has the same stories of incidents that are being telecasted. This can lead to the same interpretation of news.

- People can connect with one another and share information more easily than ever before thanks to the internet. This can result in people from various cultures learning about each other's points of view and having similar ideas on current events.

Here are some examples of how the project demonstrates how news from around the world can increase similarities:

- The word cloud reveals that the terms "ukraine" and "russia" are the most prominent in the word cloud, implying that these are the two most important topics to people worldwide like Russia-Ukraine War.

- Also, in the other word cloud, the terms "europe" and "european" are prominent in the word cloud, implying that Europeans are particularly interested in the news of the fighting in Ukraine.

The same thing can be seen in more creative way-
If news is a global village, then similarities are the town square. When news from anyplace in the world breaks, it spreads to every corner of the globe, bringing people together to share their tales and viewpoints. Similarly, this project targets this global conversation, and it demonstrates how similar our lives may be, despite our differences in culture and background.

References

1. https://www.exchange4media.com/media-tv-news/nclt-approves-proposal-for-acquisition-of-television-home-shopping-network-report-131018.html
2. https://github.com/jdepoix/youtube-transcript-api
3. https://www.techtarget.com/searchdatamanagement/definition/data-preprocessing
4. https://www.geeksforgeeks.org/python-lemmatization-with-nltk/
5. https://www.learndatasci.com/glossary/cosine-similarity/
6. https://www.turing.com/kb/guide-on-word-embeddings-in-nlp
7. https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2#background
8. https://huggingface.co/clips/mfaq