Problem:

Create a timeline summarization list, that shows the major events of the Israel Hamas war that had occurred during the time-period mentioned in the Json file

Steps followed Thought process

- Loading and preprocessing the json data.
- Taking the input from the user like “Israel-Hamas War” and generating the keywords
- Extract the date and the news article description {The news article description contains more information than the title}.
- Summarize the description using the transformer model (here, used the facebook/bart-large-cnn for text summary generation) {the description will be article and we want only relevant information not whole text as we are solving a summarization problem}.
- generate the vector embeddings for each article (i.e. description) and then form clusters of the similar articles (i.e. description) by means of some clustering algorithm (here, used DBSCAN)
- Using the keywords filter the articles based on the description and then get a central vector which represent all the filtered articles via keywords, now based on the central vector find a cluster that it belongs to base on all the data and corresponding to the vector get all the articles
- get the date corresponding to these articles and that will your results
- further summarize the articles by forming the cluster and using title or description to generate new title or description for an enhanced summary like mentioned in the problem document "An example of a time timeline summarization list is as follows (excerpt from Wikipedia):".

Let me explain the Vector embedding and clustering here in more details

First let understand the concept here

Vector Embeddings:

Convert each article description into a list of numbers (vector) that captures its meaning and important features. This process translates complex text into numerical form.

Vector Space:

Imagine each vector as a point in a high-dimensional space where similar articles are closer together, and different ones are farther apart.

Clustering (DBSCAN):

Use a clustering algorithm like DBSCAN to group these points (articles) into clusters based on their proximity. DBSCAN identifies dense regions of points and groups them together, ignoring noise (outliers).

So, in summary, convert articles into numerical vectors, place them in a high-dimensional space, and then group similar ones using DBSCAN to find clusters of related articles. This helps in organizing and summarizing the information based on their content similarity.

Let’s understand Vector Embeddings with an example:

Let's imagine you're trying to sort a big collection of toys based on how similar they are. Each toy has different features like size, color, and type (e.g., car, doll, ball). To make it easier, you can think of each toy as a point in space, where each feature (size, color, type) is a dimension. This forms the basic of data generation and generative AI.

Vector Embeddings in Simple Terms:

Mapping Toys to Points:

Each toy (or news article) is turned into a list of numbers (vector) that represents its features.

For example, a red toy car might be represented by the numbers (1, 0, 5), where 1 is its size, 0 is its type (car), and 5 is its color (red).

High-Dimensional Space:

Just like how we can use three dimensions to describe where something is in a room (length, width, height), we can use many dimensions to describe more complex features. This "space" where the toys are placed is called a vector space.

Finding Similar Toys:

Toys that are similar (e.g., two red cars) will have points that are close together in this space.

We can easily find and group similar toys by looking at the points that are near each other.

Example:

Imagine we have three toys:

- A red car: (1, 0, 5)

- A blue car: (1, 0, 3)

- A red ball: (2, 1, 5)

In our vector space:

- The red car and blue car will be close to each other because they share the same size and type.

- The red car and red ball will also be somewhat close because they share the same color.

By using vector embeddings, we can turn complex information into a form that makes it easy to compare and find similar items, just like sorting toys by their features in a big, multi-dimensional toy box.

Steps Followed in the code

Step 1: Loading and Preprocessing the JSON Data

Simple Explanation: Preprocess and clean news data to ensure it's accurate, relevant, and easy to analyze. This process removes errors, duplicates, and irrelevant information, making the data consistent and suitable for generating meaningful insights and summaries.

Note: Here very small subset of the original data is taken as the transformer model was taking very huge time to generate the summaries.

Step 2: Taking Input and Generating Keywords

Simple Explanation:

To ask the user which event he/ she would like view like “Israel Hamas War” and generate the keywords so that we can retrieve the more relevant information related to the given prompt or user input.

Example: You tell me you want to know about the "Israel-Hamas War". I think of important words like "Israel", "Hamas", "war", "conflict", etc.

Step 3: Extracting Dates and Descriptions

Simple Explanation:

We look at the stories we found and write down when they happened and a short description of what happened.

For Example: We find a story about a big event on October 7, 2023. We write down "October 7, 2023: A big event happened."

Step 4: Summarizing Descriptions

Simple Explanation:

The stories might be very long, so we use a special tool to make them shorter but still keep the important parts.

For Example: We have a long story about a speech. The tool shortens it to "A leader gave an important speech."

Step 5: Generating Vector Embeddings and Clustering

Simple Explanation:

We turn the short descriptions into special codes that a computer can understand and group similar stories together.

For Example: We make a code for each short story. Stories about speeches get one code, and stories about fights get another. Then, we put stories with similar codes together.

Step 6: Filtering Articles Using Keywords

Simple Explanation:

We use our important words to find the stories that match what we're looking for.

For Example: We look at our grouped stories and see which ones mention "Israel", "Hamas", or "war". We keep those stories.

Step 7: Finding Relevant Clusters

Simple Explanation:

We find the group of stories that best match the important words we picked earlier.

For Example: We have groups of stories. One group talks a lot about the Israel-Hamas war. We pick that group.

Step 8: Summarizing the Articles Again with timelines

Simple Explanation:

We make a final short summary of the most important stories in our chosen group with timelines.

For Example:

Final code output

```
10 September 2014 - 10 September 2014: Israeli soldier killed in rocket at
tack from Lebanon. Soldier was killed by a rocket fired from Lebanon, Isra
el says. Israeli military says it is investigating the attack and has laun
ched an investigation into the source of the rocket attack. Israel says th
e rocket came from Lebanon and was fired from inside the country.

18 December 2023 - 22 December 2023: World celebrates Christmas in shadow
of war. UN Security Council adopts resolution on aid to Gaza watered down
by US pressure. Syrians cancel Christmas festivities in solidarity with Ga
za. Pope laments war in Holy Land on solemn Christmas Eve. Pope Presides O
ver Christmas Eve Mass: 'Our Hearts Are in Bethlehem'

22 December 2023 - 25 December 2023: Israel increases strikes in central G
aza, killing scores Khawaja denied permission to have peace symbol on bat:
Reports Pope Francis says Jesus’ message of peace is being drowned out by
‘futile logic of war’ Shipping giant Maersk prepares to resume operations
in Red Sea. 50 elections globally amid wars cast doubt on 2024 economic ou
tlook.
```

Improvements:

1. Preprocess the data to remove the things like urls, new lines etc and make data cleaner.
2. Generate the import keywords using the llms for better semantic relation while searching for the relevant articles.
3. Use advance techniques like ner for related article searching.
4. For summarization and embedding generation use GPT models.
5. Use other advance techniques for finding the average or mean representation of the all documents in vector space.
6. For generating new description using the titles use GPT models.