Report

We use the OpenAI API with the DALL-E model to create the images in task 5. The script starts by loading a dataset called merged_bfro.csv into a pandas DataFrame. This dataset is expected to contain columns such as 'Id,' 'Report Type,' 'Observed,' and 'Witness Count,' among others.

The core functionality of the script is encapsulated in the process_row function, which processes each row of the dataset. For rows where the 'Report Type' is 'Report', the script checks if an image already exists for that row based on the 'Id' column. If an image exists, it skips the row to avoid redundant API calls. If no image exists, the script constructs a caption for the image using the 'Observed' and 'Witness Count' columns, with a limit of 500 characters for the 'Observed' text to ensure the caption is concise.

The caption is then used to generate an image using the generate_image_with_fallback function. This function attempts to generate an image with the full caption, and if it fails, it tries again with just the 'Observed' text as a fallback. If that also fails, it uses a generic caption related to Bigfoot sightings. The generated image is saved in the 'images' directory with a filename based on the 'Id' column.

The script uses a threading semaphore to control the rate of API requests, ensuring that it doesn't exceed the API's rate limit. The number of concurrent threads is set to 2, but this can be adjusted based on the user's needs.

Once all rows are processed, the script updates the DataFrame with the paths to the generated images and saves the updated DataFrame to a new CSV file.

From this script, you can derive a collection of images corresponding to Bigfoot sighting reports, with each image visually representing the textual description provided in the dataset. This can be useful for visualizing the data in a more engaging way or for further analysis of the sightings based on the generated images.

The Tika Docker packages for Image Captioning and Object Recognition are utilized to perform image captioning and object detection tasks on the collection of images. The detected objects and the image captions are then integrated into the dataset, enriching it with textual features for more comprehensive analysis. The process for interacting with the Tika services is split into two separate Python scripts, with the first Python script interacting with Tika's Inception Rest service to detect objects in the images, while the second script interacts with Tika's Show & Tell caption generator to produce captions for each image.

After confirming that the Docker containers have been successfully set up and are running, the Python script performing object detection is executed. It first retrieves the IP address of the system and constructs URLs to query the Tika Inception service for each image. The script then iterates through each image file, sending requests to the Tika Inception service to perform object detection. Upon receiving responses, it extracts the detected objects from the JSON data and stores them along with the image filenames and paths. Subsequently, it writes the results to a CSV file named 'object_detection_results.csv'. Afterward, it loads the dataset generated from Task 5, 'task5_dataset.csv', along with the 'object_detection_results.csv' file. The 'Image' column is dropped from the object detection results dataframe, and a left merge is performed on the 'Generated_Image_Path' column to combine the datasets. Finally, the merged dataset is saved to a new CSV file named 'merged_dataset.csv'.

Following this, a second Python script is executed to generate image captions for each image in the collection. This script is similar to the previous one but is modified to generate captions instead of performing object detection. The code iterates through each image file, sending requests to the caption generator service. Captions are extracted from the JSON responses and stored with the image filenames and paths. The results are written to a CSV file named 'image_captions.csv'. After generating the image captions, the script loads the previously merged dataset, 'merged_dataset.csv', and the 'image_captions.csv' file. The datasets are combined, and the resulting merged dataset is stored in a new CSV file named 'task6_dataset.csv'. These Python scripts utilize Tika Docker files to enrich the dataset with object detection results and image captions, facilitating further analysis.

All generated object detections and image captions were retained, resulting in each image having a list of object detections and a list of image captions. The decision to keep and consolidate all captions and detected objects into lists was driven by the goal of preserving all generated information, thereby offering a comprehensive overview of potential image descriptions and object detections for each image. This aggregation of text enables flexibility in analyzing and exploring the data, providing access to multiple potential descriptions and object detections linked with each image.

Although image captions and object detections were successfully generated, they exhibited inaccuracies. The image captions often fail to accurately depict the images, frequently missing the mark or providing descriptions unrelated to the visuals in the images. Even when there is partial agreement between the captions and the depicted scenes in the images, discrepancies persist, with descriptions sometimes being slightly off. Notably, there is a tendency for captions to reference people skiing on snow-covered slopes, even when such activities are not depicted in the images. Despite the presence of mountains or snow in some images, individuals may not always be skiing or may not even be situated on the mountain slopes, but are rather situated in a valley between the mountains. Additionally, the image captions often misidentify entities presumed to be Bigfoot, erroneously categorizing them as humans, bears, dogs, or other animals.

The identified objects also often diverge from the details reported in the original sightings. While a few terms in the detected objects list may align with some of the details reported in the sightings, a significant portion of the words in the identified objects lists are unrelated to the original observations. There is also often a discrepancy between identified objects and the image captions, with minimal overlap observed between the two. For instance, the list of detected objects for an image may contain terms such as "website" or "comic book," while the corresponding captions describe individuals skiing on snow-covered mountains. Similarly, in another image, the detected objects may reference apes or dogs, while the captions mention bears instead. Although occasional agreement between the identified objects and the generated captions occurs, it typically involves only a few shared terms, with the remainder of the identified objects not reflected in the generated image captions.

Notable patterns observed in the text captions include frequent descriptions of individuals or animals positioned on hills or mountains, even when said hills or mountains are absent from the images, or when the subjects are clearly not situated on elevated terrain. Furthermore, a tendency towards repetition is evident, with captions often conveying similar messages, albeit with minor variations in wording or sentence structure. Moreover, many text captions feature the phrase "a group of people." Additionally, the inclusion of references to animals such as dogs, bears, or apes is recurrent, presumably alluding to the depictions of Bigfoot within the images.

Meanwhile, notable trends observed in the detected objects include frequent occurrences of repetitive terms with minor variations in the spelling, such as "website" and "web site", or closely related words like "lakeshore" and "lakeside". Another noteworthy pattern involves the inclusion of scientific animal names, such as those of various primate or sheep species, in the 'Detected Object(s)' column. Furthermore, inconsistencies arise when certain detected objects do not visually match with what is depicted in the actual images or are absent from the original sighting reports. Moreover, another notable trend involves the prevalence of words related to apes or monkeys in the objects detected column, contrasting with the more frequent occurrence of words related to bears and dogs in the image captions column.

Despite the discrepancies with the object detections and image captions compared to the original sightings, using the Tika Image Captioning and Object Recognition services proved to be beneficial in enriching the dataset and uncovering insights that are potentially valuable for future analysis.

Extracting precise location information from unstructured text data is a non-trivial task requiring sophisticated natural language processing techniques. In the context of the BFRO sightings dataset, the need to accurately identify and geocode location entities mentioned in the textual descriptions became paramount for enabling further analysis and exploration of potential geographical patterns and correlations.

The Tika GeoTopicParser, a powerful NLP tool developed by the Apache Software Foundation, was employed to accomplish task 7. The parser leverages a combination of machine learning algorithms and domain-specific gazetteers to recognize and extract location names, along with their corresponding latitude and longitude coordinates, from unstructured text inputs.

Initially, multiple text fields from the dataset, including 'State,' 'County,' 'Location Details,' 'Nearest Town,' and 'Nearest Road,' were concatenated to construct comprehensive location descriptions for parsing. However, it soon became evident that many of these fields contained missing or incomplete data, leading to gaps in the extracted location information. The 'State' field was utilized as a fallback option to mitigate this issue, as it generally contained more complete and reliable data.

By running the GeoTopicParser on the state names, it was possible to populate the 'Location,' 'Longitude,' and 'Latitude' columns with reasonable approximations where other location details were unavailable. While this approach may not always reflect the precise locations described in the sightings, it provided a valuable starting point for further analysis, such as mapping and identifying potential hotspots or patterns based on geographical coordinates.

It is worth noting that the effectiveness of the Tika GeoTopicParser, like any NLP tool, is inherently dependent on the quality and completeness of the input data. In cases where location details were sparse or ambiguous, the parser's ability to accurately identify and geocode the entities was diminished. Nonetheless, the tool proved to be a powerful asset in the data enrichment process, enabling the extraction of location information from unstructured text with a high degree of accuracy where suitable inputs were provided.

Overall, the process of extracting location data from the BFRO sightings dataset using the Tika GeoTopicParser was a valuable exercise in leveraging state-of-the-art NLP techniques for data enrichment. While the results may not be perfect, they provide a solid foundation for further analysis and exploration of potential correlations between sightings and geographical factors, ultimately contributing to a deeper understanding of the phenomenon under investigation.

Our exploration of the Bigfoot Sighting Dataset involved a comprehensive analysis utilizing SpaCy, a robust natural language processing library. By employing SpaCy, we were able to delve into the textual descriptions provided in various sections of the dataset, extracting valuable insights related to geographical locations, temporal details, environmental settings, detected objects, and image captions associated with Bigfoot sightings.

One of the key aspects we focused on was the extraction of named entities, which provided a wealth of information regarding the locations where Bigfoot sightings were reported. Through SpaCy, we identified prominent geographical entities such as "Anchorage," "Potter Marsh," "Fairbanks," and "Wrangell - St. Elias National Park," hinting at specific regions where encounters with Bigfoot are prevalent. Notably, the mention of national parks like "Wrangell - St. Elias National Park" suggests that these areas may serve as significant habitats or migration routes for the elusive creature.

Temporal entities extracted from the dataset offered valuable insights into the timing of Bigfoot sightings. Phrases like "the middle of the night," "Approximately 12:30 pm," and "Early morning" provided context regarding the timeframes during which these encounters occurred. Such temporal information is crucial for understanding the behavioral patterns of Bigfoot and may help researchers identify potential patterns or trends associated with sightings.

Furthermore, our analysis uncovered environmental details that shed light on the habitats and surroundings where Bigfoot encounters took place. Mentions of locations such as "Campbell Creek," "Anchorage Coastal Wildlife Refuge," and "Goldstream Valley" provided insights into the ecological settings favored by Bigfoot. Additionally, detected objects mentioned in the dataset, although sparse, offered supplementary context to the sightings, potentially indicating interactions between Bigfoot and its surroundings.

However, it's important to note that while the textual descriptions provided valuable information, the image caption section of the dataset appeared to lack entities, highlighting a potential area for improvement in data collection or processing. Integrating image analysis techniques alongside textual analysis could provide a more holistic understanding of Bigfoot sightings and help corroborate the information extracted from textual descriptions.

In conclusion, our in-depth analysis using SpaCy unveiled a myriad of insights into Bigfoot sightings, ranging from geographical distributions and temporal patterns to environmental contexts and detected objects. Moving forward, further research efforts could focus on refining entity extraction techniques, incorporating image analysis methods, and collaborating with domain experts to deepen our understanding of the mysterious phenomenon surrounding Bigfoot sightings.

I find using spaCy both easy and challenging. One relatively easy aspect is that it provides pre-trained models. These models can be easily loaded and used for a variety of NLP tasks such as named entity recognition (NER) and part-of-speech tagging (POS). This makes it easy to get started with NLP tasks without having to train a model from scratch, saving time and computing resources.

Another advantage of spaCy is its well-designed and intuitive API. spaCy has a clear and consistent interface that allows users to perform text processing tasks such as tokenization, sentence segmentation, and entity recognition with just a few lines of code. It helps novices like me read and understand the content intuitively.

However, training and using the deep learning models provided by spaCy can be resource-intensive, requiring powerful hardware (e.g. GPU) and large amounts of memory. This can be an obstacle for users with limited computing resources. Additionally, mastering all of spaCy's features and understanding how to utilize them effectively can take time. For me, spaCy is a completely unfamiliar thing, which took me a lot of time to learn.

Overall, while spaCy greatly simplifies many aspects of natural language processing, it still requires a solid understanding of machine learning principles and NLP concepts to use effectively.

While the powerful capabilities of GeoTopicParser are undeniable, the actual process of using them was not always smooth sailing. Firstly, the tool's compatibility with the Windows operating system left much to be desired, requiring additional configurations or the use of virtual machines. Secondly, the installation process itself presented several hurdles, especially when it came to testing out the GeoTopicParser. The documentation provided for this step was not particularly detailed, and it took a significant amount of trial and error before we were finally able to run the parser successfully. Overall, despite the immense value offered by these machine learning and deep learning tools, the process of actually putting them into practice was not always straightforward, highlighting the need for developers to continue optimizing.

Running the Tika Docker services also presented some challenges, as there was some initial confusion during the setup process. Even when following along with guidelines written out to help with the process of installing and running the Dockers, there was still some trial and error when building and running the Dockers. Additionally, there was some confusion about which IP address to use when running the curl command in PowerShell when making an HTTP GET request to a specific endpoint URL, which was part of the process of working with the Tika Docker images.

Another drawback of using Tika Dockers was the significant time required to generate captions and detect objects across the image collection. There was a substantial volume of images requiring caption generation and object detection, but it still took several hours to execute the Python scripts and interact with the Tika Docker services to retrieve image captions and object detections. The necessity of downloading additional software or tools, such as Git and Node.js, to work with the Tika Docker images also came as a surprise. This unexpected requirement lengthened the process of successfully setting up the Tika Docker services.

After overcoming the installation hurdles and successfully launching the Docker containers, utilizing the Tika Docker services became mostly straightforward. However, trying to run two different Docker containers at once posed a challenge, and in the end it could not be done, so it was decided to just run one container at a time. Running only one Docker container at a time required dividing the code into two different Python scripts, and this contributed to a longer runtime when executing the code for Task 6.