

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | Article Sentiment and Entity Extraction | | | | |  | |
|  |  | | | | | | |  |
|  | | | |  |  | | | |
|  | | | | Dharani Sai Chetna |  | | | |
|  | | | | December 16, 2024—Data Science Assessment—Intelliwings |  | | | |
|  | | |  | | |  | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | |  |  | | |  |
|  | INTRODUCTION | | | | | | |  |
|  |  | | |  |  | | |  |
|  |  | |  | | |  | |  |
|  |  |  | The objective of this project was to build an automated system that could analyze online articles by extracting key information and determining their overall sentiment. The system performs two main tasks:   1. Entity Extraction: Identifying and extracting key entities such as people, organizations, locations, and other significant elements within the article. In this case, the focus was primarily on extracting PERSON (individuals) and ORG (organizations) entities. 2. Sentiment Analysis: Analyzing the overall sentiment of the article, categorizing it as positive, negative, or neutral. This is done by processing the text and using a sentiment analysis model that outputs a sentiment label along with a confidence score indicating the certainty of the prediction.   This solution leverages several modern technologies, including web scraping, Natural Language Processing (NLP), and sentiment analysis. The web scraping part is done using the requests and BeautifulSoup libraries, while NLP tasks, including entity extraction, are handled by the spaCy library. Sentiment analysis is performed using a pre-trained model from Hugging Face's transformers library. | | |  |  |  |
|  | | | | |
|  |  |  |  |
|  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Decorative | |  |  | | |  |  | |
|  | | APPROACH | | | | |  | |
|  | |  |  | | |  |  | |
|  | To achieve the desired functionality, I followed a systematic approach that included four main steps:   1. **Web Scraping**: The first part of the solution involved fetching the article content from a given URL. Using Python's requests library, I sent HTTP requests to the URL to fetch the HTML content of the article. Once the content was retrieved, I used BeautifulSoup to parse the HTML and extract the relevant article body. I focused on locating the article within either the <article> or <body> tags, depending on the structure of the website. This allowed me to get the core textual content that was ready for further analysis. 2. **Entity Extraction**: After scraping the article content, I processed the extracted text using spaCy, a popular NLP library. I used the pre-trained en\_core\_web\_sm model, which is specifically designed for general-purpose English text analysis. Once the text was processed, spaCy automatically identified named entities such as persons, organizations, locations, and other proper nouns. For this particular project, I focused on extracting PERSON and ORG entities, which are the most relevant for identifying key people and organizations mentioned in the article. The extracted entities were stored as pairs of entity name and type (e.g., "Elon Musk" - PERSON). 3. **Sentiment Analysis:** The next step was to analyze the sentiment of the article. I used the pre-trained sentiment analysis model from Hugging Face, specifically the distilbert-base-uncased-finetuned-sst-2-english model, which is fine-tuned for sentiment classification tasks. This model classifies the sentiment of the article into three categories: positive, negative, and neutral. To handle potential issues with long text (since many articles exceed the token limit of 512 characters), I truncated the article to 512 characters before passing it to the model for analysis. The sentiment analysis result includes both the predicted sentiment label and a confidence score, which indicates how confident the model is in its classification. 4. **Database Storage:** Once the article’s entities and sentiment were extracted, the final step was to store this information in a database for future reference. I used SQLite, a lightweight database system, to create a table that stores each article's URL, content, extracted entities, and sentiment. This not only allows easy access to the processed data but also helps track multiple articles for further analysis and comparison. | | | | | | |  |
|  |  | | |  |  | | |  |

**Challenges Faced**

During the development of this project, I encountered several challenges that tested my problem-solving skills and understanding of the various technologies involved:

1. **Scraping Dynamic Content:** Some websites load their content dynamically using JavaScript. This posed a challenge since BeautifulSoup and requests only work with static HTML content. As a result, I had to focus on scraping parts of the page that were less likely to be dynamically generated, such as <article> and <body> sections. While this worked for most articles, some websites required more complex solutions, such as using headless browsers (e.g., Selenium), to scrape content that was rendered dynamically.
2. **Entity Extraction Accuracy:** Although spaCy provides excellent performance for entity extraction, I found that it struggled with recognizing certain entities, especially less common ones or those that are presented in less conventional ways. For example, if an article mentions an organization by a nickname or a less formal name, the model might miss it. Moreover, spaCy often did not capture entities that were mentioned indirectly or contextually, such as “the CEO of Tesla” instead of “Elon Musk.” This meant that some important entities might have been overlooked.
3. **Sentiment Analysis Limitations:** The sentiment analysis model from Hugging Face worked well for straightforward articles, but it had difficulty with articles that contained mixed sentiments. For example, an article that praises a company but also points out its flaws could be categorized as overly negative because the model might only pick up on the critical tone. Additionally, the truncation of text to fit within the 512-token limit sometimes resulted in incomplete sentiment analysis, especially for longer articles where important context might have been lost.
4. **Handling Long Articles:** As mentioned, long articles often exceeded the token limit of 512 characters for sentiment analysis. To handle this, I truncated the text to the first 512 characters, but this approach is far from ideal. A more advanced method would involve chunking the article into smaller parts and analyzing each chunk separately, then aggregating the results to provide a more accurate sentiment classification.

|  |  |
| --- | --- |
|  |  |

**Reflections on Accuracy**

1. **Entity Extraction:** The entity extraction process was mostly successful, with spaCy effectively identifying well-known entities such as famous persons and companies. However, there were instances where less familiar entities were missed or misclassified. Fine-tuning the entity recognition model on specific datasets or incorporating additional entity recognition techniques could improve accuracy. For instance, a custom-trained model might be more effective in recognizing domain-specific entities that are relevant to particular industries or topics.
2. **Sentiment Analysis:** The sentiment analysis model performed well on articles with clear sentiment, either positive or negative. However, it faced challenges with articles that were more nuanced or complex. Articles that included mixed emotions (e.g., both praise and criticism) often got categorized as having a single dominant sentiment, which did not always reflect the true tone of the article. One potential solution is to use a more advanced or multi-class sentiment analysis model that can detect subtler tones, such as neutral sentiment or mixed emotions. Additionally, incorporating a more context-aware model could lead to better results.

**Conclusion**

This project successfully demonstrated how web scraping, entity extraction, and sentiment analysis can be combined to analyze online articles. It showed that automated systems can effectively process textual data, identify key entities, and evaluate the sentiment of articles, providing valuable insights into the content.

While the solution worked well for most use cases, there were several areas for improvement. Fine-tuning the models used for both entity extraction and sentiment analysis could enhance their accuracy. More sophisticated scraping techniques and a deeper understanding of article context would also help improve the results.

In future iterations, I plan to incorporate more advanced NLP models, such as those designed specifically for sentiment analysis with mixed emotions, and enhance the entity extraction process by adding more contextual data. By addressing these limitations, the system could be made more robust and capable of handling a broader range of articles with greater accuracy.

This project provided valuable experience in working with a variety of technologies and frameworks. The solution presented a functional and efficient way of analyzing articles, and with some improvements, it could serve as a powerful tool for content analysis in areas like media monitoring, customer feedback, and social media sentiment analysis.