Ruben Valenzuela

Miguel Mora  
Professor Anna Devarakonda

Artificial Intel Applications

28 July 2025

**Reflection Journal – Final Project**

**1. Introduction**

The objective of this final project was to transform our midterm NewsBot, a basic text classification model, into a comprehensive, production-ready news analysis platform integrating advanced natural language processing techniques. The goal was to demonstrate mastery of the entire NLP pipeline, covering everything from data preprocessing and classification to sentiment analysis, named entity recognition, multilingual capabilities, and content generation. Through this process, we aimed to build a modular and extensible system that not only classifies news articles accurately, but also extracts meaningful insights from unstructured text data, providing a valuable proof of concept for real-world applications.

**2. Our NLP Pipeline and System Architecture**

Our final system was structured as a multi-stage NLP pipeline, with each phase corresponding to a core concept taught throughout the course. The design was modular, allowing independent testing and integration of each component. The main stages included:

* **Data Preprocessing:** We began with a comprehensive text cleaning and normalization phase. Using nltk and spaCy, we performed tokenization, stopword removal, and lemmatization. We also removed punctuation, numerical values, and excess whitespace to ensure optimal feature extraction.
* **TF-IDF Vectorization and Term Analysis:** We used scikit-learn’s TfidfVectorizer to convert our corpus (BBC News dataset with 1,490 articles) into a numerical format. We analyzed the most informative terms per category and visualized key features, which helped us interpret the model’s decisions and gain insight into class-specific language.
* **Text Classification:** A Multinomial Naive Bayes model was trained on the vectorized dataset, achieving an accuracy of approximately **97%**. We computed precision, recall, and F1-score per class and created a confusion matrix to assess model performance. The classifier was integrated into a command-line interface for interactive use.
* **Sentiment Analysis:** Using the VADER sentiment analyzer, we extended our system to assess the emotional tone of news articles. Each article received a compound score, and we labeled it as positive, neutral, or negative. We visualized the sentiment distribution across categories to identify patterns in tone.
* **Named Entity Recognition (NER):** With spaCy, we extracted named entities such as PERSON, ORG, and GPE. This added a semantic layer to our analysis and allowed us to investigate the most frequently mentioned entities within the dataset.
* **Dependency Parsing:** We employed spaCy’s syntactic parser and rendered dependency trees using displacy. This helped us understand the grammatical structure of news articles and revealed the hierarchical relationships between words.
* **Text Summarization:** We integrated the t5-small Transformer model from Hugging Face to generate abstractive summaries of long articles. This functionality adds significant value for users seeking quick insights and demonstrates our understanding of language generation models.
* **Multilingual Support:** We implemented automatic language detection using langdetect and incorporated googletrans to translate non-English articles into English before classification and summarization. This feature greatly expanded the accessibility and versatility of our system.

**3. Challenges and Problem-Solving Strategies**

Throughout development, we encountered multiple technical and architectural challenges that required careful planning and problem-solving:

* **Library Conflicts and Versioning Issues:** The integration of googletrans introduced conflicts with httpx, especially when attempting to use Gradio. We resolved this by isolating translation logic and avoiding incompatible UI frameworks within Colab.
* **Model Performance vs. Runtime:** When working with Transformer-based models like T5, we needed to balance model performance with memory and runtime limitations in Google Colab. We used smaller models (e.g., t5-small) and truncated inputs to manage GPU constraints effectively.
* **Multilingual Token Normalization:** Translating text before applying tokenization and lemmatization required careful sequencing of preprocessing steps. We ensured that translations occurred early in the pipeline to avoid errors downstream.
* **User Interface Integration:** Although we intended to implement a Gradio interface for bonus points, we encountered transport-related async conflicts that made full integration difficult. To prioritize system stability and pipeline performance, we documented the issue and left the interface as a future improvement.

**4. Key Takeaways and Lessons Learned**

This project provided us with an end-to-end, hands-on experience in building a real-world NLP application. We deepened our understanding of core concepts including vectorization, classification, language modeling, and multilingual support. We also developed practical skills in:

* Debugging large NLP systems across multiple libraries
* Structuring modular and scalable code
* Designing effective data visualizations
* Using APIs and pretrained models responsibly
* Presenting technical work to non-technical audiences

Additionally, this project highlighted the importance of ethical considerations in language generation, transparency in model interpretation, and the potential of NLP to solve complex real-world problems.

**5. Future Work and Improvements**

While our system is already functional and comprehensive, there are several areas we would like to improve or expand in future iterations:

* **Semantic Search Engine:** Add a search feature to find semantically similar articles using sentence embeddings.
* **UI Deployment:** Finish implementing a Gradio or Streamlit interface with full translation + classification flow.
* **Named Entity Aggregation:** Visualize entity networks and relationships across articles (e.g., co-occurrence graphs).
* **Live News API Integration:** Allow real-time ingestion of news headlines via RSS feeds or APIs.
* **Bias Detection and Ethical Analysis:** Incorporate tools to analyze political or emotional bias in reporting.

**6. Team Roles and Contributions**

We collaborated effectively throughout the project, dividing responsibilities based on our individual strengths and sharing knowledge consistently. Below is a breakdown of our contributions:

**Ruben Valenzuela:**

* Designed and implemented the text preprocessing pipeline
* Developed sentiment analysis, NER, and dependency parsing modules
* Integrated text summarization with Transformers
* Built multilingual detection and translation functionality
* Final testing, code cleanup, and visualization formatting

**Miguel Mora:**

* Created and optimized the TF-IDF classification model
* Conducted term frequency analysis and class-specific feature visualization
* Evaluated model accuracy and built confusion matrix
* Assisted with initial interface logic and repository organization
* Drafted documentation outlines and reviewed final integration

We are proud of what we built and excited about the potential of NewsBot as a foundation for future NLP projects.

Github: https://github.com/Rubenvalenzuelaaa/newsbot-intelligent-classifier/blob/main/README.md