MACQUARIE University
SYDNEY·AUSTRALIA

Unit: Advanced Natural Language Processing,
COMP8420

Submitted by: Sahar Tosif
Student Id: 47536519
Email: sahartosif.sahartosif@students.mq.edu.au

And

Submitted by: Mahak Mahak
Student ID: 47552840
Email: mahak.mahak2@students.mq.edu.au

Professor: Qiongkai Xu, Greg Baker
Tutor: Congbo Ma

News Recommender System

Project Report

Introduction:

In the modern digital age, the sheer volume of available news content can be overwhelming for readers. The constant influx of information makes it difficult for users to find news articles that are relevant to their interests, leading to information overload, decision-making difficulties, and reduced attention spans.

Our project addresses this real-world challenge by developing a news recommender system that enhances user experience by delivering tailored news articles. The goal is to increase user engagement and satisfaction by providing content that aligns with individual preferences and interests. By leveraging advanced Natural Language Processing (NLP) techniques, our system aims to overcome the problem of information overload and ensure that users receive timely, relevant, and diverse news content.

Utilising the AG News dataset from Kaggle, which includes a diverse collection of news articles categorised into World, Sports, Business, and Sci/Tech, our system analyses user queries and article content to recommend the most pertinent articles. This not only helps users stay informed about topics that matter to them but also improves their overall news consumption experience by filtering out irrelevant information.

In summary, our personalised news recommender system addresses the significant challenge of navigating vast amounts of online news content, ultimately aiming to provide a more engaging, efficient, and relevant news reading experience for users.
The complete project, including the source code and detailed documentation, is available on our Github Repository.

Methodology:

Dataset Utilised:

In the news recommender system project, we utilised the AG News Topic Classification Dataset, a comprehensive collection of over 1 million news articles. This dataset was sourced from more than 2,000 news sources over the span of a year by ComeToMyHead, an academic news search engine operational since July 2004. The dataset is specifically designed for research purposes in various fields such as data mining, information retrieval, and data compression, making it an ideal choice for our project.

The AG News dataset is constructed by selecting the four largest classes from the original corpus, each class comprising 30,000 training samples and 1,900 testing samples. This results in a total of 120,000 training samples and 7,600 testing samples, providing a robust foundation for training and evaluating our news recommender system.

The dataset is organised into two main files:

- **Train.csv:** Contains the training samples with 3 columns - class index (ranging from 1 to 4), title, and description.
- **Test.csv:** Contains the testing samples with the same structure as the training file.

Each row in the train.csv and test.csv files is formatted as comma-separated values with the title and description enclosed in double quotes. Internal double quotes are escaped by two double quotes, and new lines within the text are escaped by a backslash followed by an 'n' character.

This well-structured and extensive dataset not only provides a diverse set of news articles for training and evaluation but also ensures a balanced distribution across different news categories, making it an excellent choice for developing a robust and accurate news recommender system.

Technologies and Evaluation Methods:

Technologies Implemented for Data Cleaning:

For data cleaning, we utilised several key technologies. The NLTK (Natural Language Toolkit) library provided essential tools for text processing, including stopword removal, tokenization, and lemmatization, which help in eliminating common insignificant words, breaking down text into individual tokens, and reducing words to their base forms, respectively. Regular expressions (re) were employed to identify and remove URLs and detect and eliminate publisher names at the start of the text. Additionally, standard Python string operations were used to remove punctuation and convert text to lowercase, ensuring uniformity and consistency. By leveraging these technologies, the data cleaning process effectively standardises and prepares text data for subsequent NLP tasks, enhancing both accuracy and efficiency.

Technologies Implemented for Keyword Extraction:

- **scikit-learn:** Utilised for its `TfidfVectorizer` to convert text into numerical feature vectors.
- **TF-IDF Vectorization:** Captures the importance of words by considering their frequency in the document and their rarity across the entire corpus, helping identify significant keywords.
- **Stop Words Removal:** Configured to exclude common stop words, ensuring only meaningful words are considered.
- **Python Exception Handling:** Implements try-except blocks to handle potential errors during vectorization, ensuring robustness.

These technologies ensure the extraction of relevant keywords from the text, enhancing the overall effectiveness of the recommendation system.

Technologies Implemented for Text Classification:

We implemented and evaluated two different models for text classification to identify the most effective approach. After a thorough evaluation, we selected the model that demonstrated superior performance.

We selected these models because we wanted to check how differently a traditional and an advanced NLP model will perform the task on the dataset.

Model 1: TF-IDF and Logistic Regression:

TF-IDF (Term Frequency-Inverse Document Frequency): We used `TfidfVectorizer` from scikit-learn to convert text data into numerical feature vectors. This method captures the importance of words in the context of the documents they appear in, considering both their frequency in the document and their rarity across the entire corpus.
Logistic Regression: Logistic Regression, was employed to classify the news articles. This model is known for its simplicity and effectiveness in binary and multi-class classification tasks.

Model 2: BERT-based Model:

BERT (Bidirectional Encoder Representations from Transformers): We utilised `AutoTokenizer` and `AutoModelForSequenceClassification` from the Hugging Face Transformers library. BERT is a state-of-the-art transformer model pre-trained on a large corpus and fine-tuned for text classification tasks.
Tokenization: The BERT tokenizer converts the text data into tokens that the model can process, ensuring the text is in a suitable format for the model.
PyTorch: Used for handling the dataset and model training. PyTorch tensors represent and manipulate data during the training and evaluation processes.

Evaluation Method for Text-Classification:

To assess the performance of our text classification models, we utilised standard classification metrics: precision, recall, and F1-score. These metrics provide a comprehensive understanding of how well our models can classify news articles into predefined categories. The evaluation was conducted on both the validation and test sets to ensure the robustness of our results.

From the table below we can observe the results indicate that the BERT model provides superior performance in terms of precision, recall, F1-score, and accuracy. Specifically, the BERT model achieved a test accuracy of 0.91 compared to 0.89 for the Logistic Regression model. This demonstrates that the BERT model is more effective in accurately classifying news articles into predefined categories. Therefore, we chose the BERT-based model for our news recommender system to ensure higher accuracy and relevance in the recommendations provided to users.

Experimental Results

Model	**Dataset**	**f1-score**	**Recall**	**Precision**	**Accuracy**
Logistic regression	validation	0.89	0.89	0.89	0.89

	test	0.89	0.89	0.89	0.89
BERT	Validation	0.90	0.90	0.90	0.90
	test	0.91	0.91	0.91	0.91

Technologies Implemented for Text Summarisation:

Since the dataset is too large, summarising each row would take approximately two days. Therefore, we will take random sample data from the dataset and summarise it in the demo that we have given in the .ipynb file.

In our project, we implemented and evaluated two different models for text summarisation to determine the most effective approach. After thorough evaluation, we selected the model that demonstrated superior performance.

Model 1: DistilBART:

DistilBART: A distilled version of the BART (Bidirectional and Auto-Regressive Transformers) model, which is a state-of-the-art transformer model designed for sequence-to-sequence tasks such as summarisation.
Tokenization: The `BartTokenizer` from the Hugging Face Transformers library is used to tokenize the input text.
summarisation: The `BartForConditionalGeneration` model generates concise summaries by encoding the input text and then decoding it to produce a summary.

Model 2: T5 (Text-to-Text Transfer Transformer):

T5: A versatile transformer model that treats every NLP problem as a text-to-text problem, including summarisation.
Tokenization: The `T5 Tokenizer` from the Hugging Face Transformers library is used to tokenize the input text.
summarisation: The `T5ForConditionalGeneration` model generates summaries by encoding the input text and then decoding it to produce a summary.

Evaluation Method for Summarisation:

To evaluate the performance of our text summarisation models, we used the BLEU (Bilingual Evaluation Understudy) score. We first generated reference summaries using the TextRank algorithm and then compared the summaries generated by our models against these reference summaries.
The BLEU score measures the similarity between the generated summary and a reference summary by comparing the n-grams (contiguous sequences of n words) in the generated and reference texts. It is a widely used metric for evaluating machine-generated text in tasks such as translation and summarisation.

Upon comparing the results, the DistilBART model demonstrated better performance with an average BLEU score of 0.5605, while the T5 model had an average BLEU score of 0.4266. This indicates that the summaries generated by the DistilBART model were more similar to the reference summaries, making it the better choice for our news recommender system. Therefore, we selected the DistilBART model for summarising news articles to ensure higher accuracy and relevance in the generated summaries.

Experimental Results

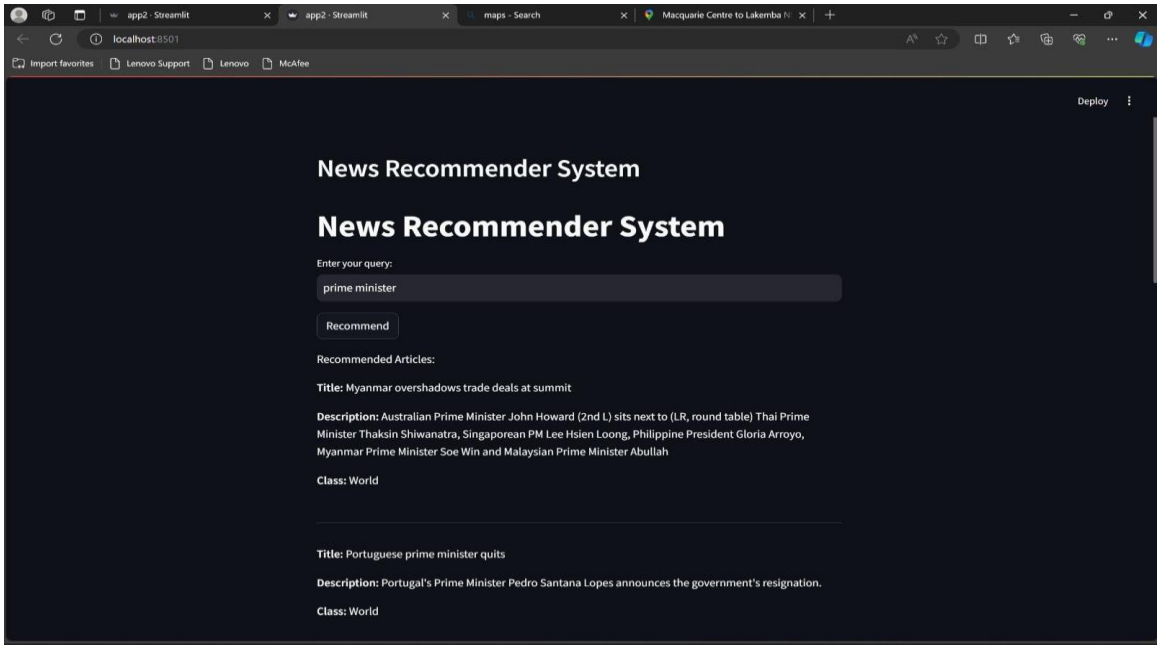
Model	**Average BLEU Score**
DistilBART	0.5605
T5	0.4266

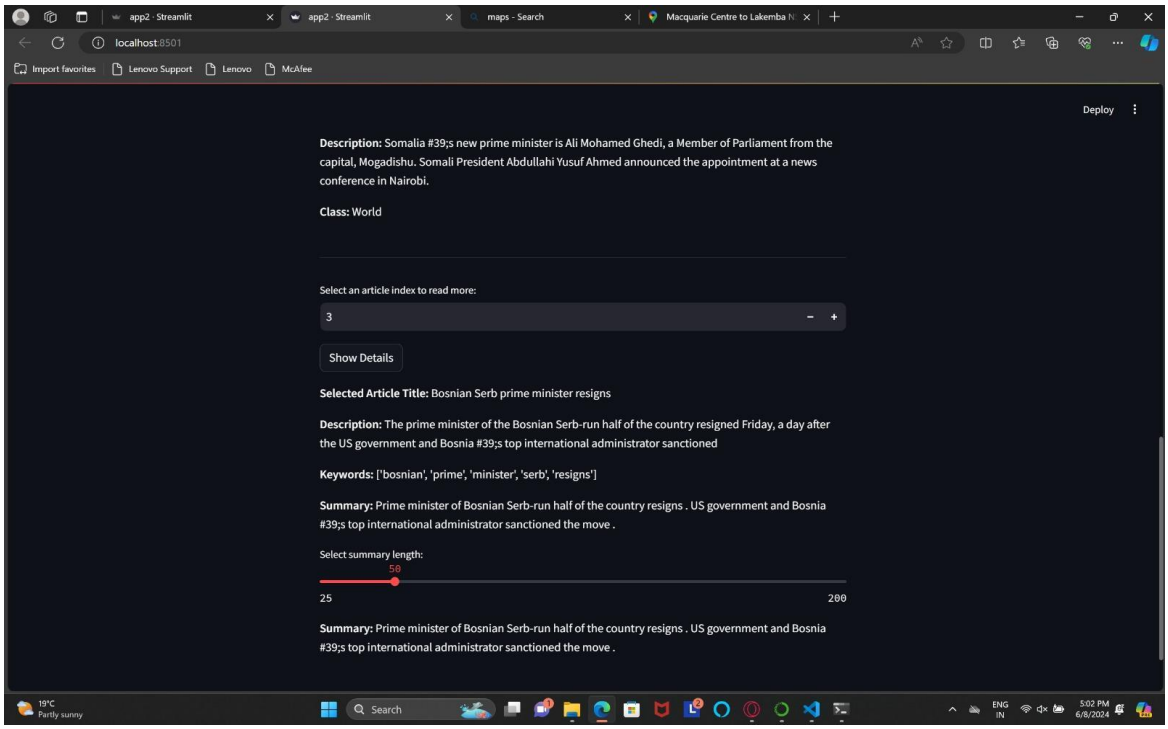
Technologies Implemented for Article Recommendation

TF-IDF Vectorization: The text data from the news articles is transformed into TF-IDF feature vectors using `TfidfVectorizer`. This involves fitting the vectorizer to the entire dataset and transforming the text into numerical vectors, which represent the importance of words within the documents.
Query Transformation: The user query is transformed into a TF-IDF vector using the same `TfidfVectorizer`.
Cosine Similarity Calculation: The cosine similarity between the query vector and the dataset vectors is calculated. This similarity measure helps identify the articles that are most similar to the user's query.
Recommendation Generation: The articles are ranked based on their cosine similarity scores, and the top 5 articles are recommended to the user. The titles and descriptions of the recommended articles are then printed.

Application Implementation

Our news recommender system features a Streamlit-based web application where users input their queries to receive personalised news recommendations. Developed using Streamlit for the frontend, the application leverages NLTK and scikit-learn for text preprocessing and feature extraction, while advanced NLP models like BERT and BARTfrom Hugging Face are used for classification and summarization. Data manipulation is handled by Pandas and NumPy. The workflow involves preprocessing the user query with NLTK and transforming it with Tfidf Vectorizer, then predicting the class using BERT. Cosine similarity is calculated between the query and article vectors to find the most relevant articles, which are then displayed with their titles, descriptions, and predicted categories. Moreover, it also allows the user to select the length of the summary.





Conclusion:

In this project, we developed a comprehensive news recommender system to enhance user experience by providing tailored news articles. Addressing the challenge of information overload, our system leverages advanced Natural Language Processing (NLP) techniques to deliver relevant, timely, and diverse news content.

Using the AG News dataset from Kaggle, we implemented and evaluated various models and methodologies for data cleaning, keyword extraction, text classification, text summarisation, and article recommendation. Each component ensures the system's accuracy and efficiency.

For text classification, the BERT-based model outperformed the TF-IDF with the Logistic Regression model, making it our preferred choice. For text summarisation, the DistilBART model demonstrated superior performance over the T5 model based on the BLEU score. Our recommendation system employs TF-IDF vectorization and cosine similarity to rank and recommend the most relevant articles based on user queries.
Lastly, the Streamlit-based web application developed for the personalised news recommender system provides an intuitive interface for users to input their queries and receive relevant news article recommendations. By leveraging advanced NLP models like BERT, DistilBart and implementing robust data processing and recommendation algorithms, the application effectively addresses the challenge of information overload and enhances the news reading experience.

In summary, our news recommender system effectively navigates vast amounts of online news content. By integrating state-of-the-art NLP techniques, we provide users with a more engaging, efficient, and relevant news reading experience, enhancing user satisfaction and engagement.
For more details, source code, and additional resources, please visit our Github Repository.

Contribution

In this project, Sahar and Mahak collaboratively developed a news recommender system to enhance user experience by delivering personalised news articles. The roles and workload distribution were as follows:

Project Planning and Coordination: Sahar led the planning phase, defining the project's scope, objectives, and timeline. Both Sahar and Mahak ensured that the project stayed on track and met the deadlines.
Data Preparation and Cleaning: Sahar was responsible for finding the right dataset that met the requirements of the project and Mahak worked on cleaning, and preprocessing the AG News dataset. This involved handling missing data, normalising text, and preparing the dataset for model training.
Model Training and Evaluation:

- **Logistic Regression and TF-IDF:** Mahak trained the logistic regression classifier and TF-IDF vectorizer to establish a baseline model for news classification.
- **BERT:** Sahar implemented the BERT model for classification, leveraging its advanced capabilities for understanding the context and semantics of news articles.
- **BART and T5:** Sahar explored the use of BART and Mahak worked on the T5 model for text summarisation, integrating them into the system to enhance its performance.

Backend Development:

- **Recommendation Algorithm:** Mahak developed the backend logic for the recommendation algorithm, including the calculation of cosine similarities between user queries and article embeddings.
- **Class Predictions:** Sahar implemented the class prediction functionality using the trained models, ensuring that the system could accurately categorise news articles.

Evaluation and Optimization: Mahak evaluated the performance of different models, comparing their accuracy and efficiency. Sahar optimised the models and algorithms to improve the overall performance of the recommender system.

Frontend Development:

- **User Interface Design and Implementation:** Sahar designed an intuitive and user-friendly interface for the recommender system, ensuring that users could easily interact with the application.

Testing and Debugging:

- **Unit Testing**: Both Mahak and Sahar performed extensive unit testing to ensure the correctness of individual components.
- **System Testing**: Both Mahak and Sahar conducted comprehensive system testing, identifying and fixing bugs to ensure the application functions correctly and efficiently under various scenarios.
- **User Interface Testing:** Mahak had tested the application, ensuring it worked well for any user.

Documentation:

- **Code Documentation**: Sahar documented the codebase, providing clear and concise explanations of the functions and classes used in the project.
- **User Guide**: Mahak created a user guide to help end-users understand how to use the recommender system.
- **Technical Report**: Mahak contributed to the technical report, detailing the methodology, implementation, results, and discussion.

Model Integration: Both Sahar and Mahak worked together to integrate BERT, BART, and T5 models into the recommender system, ensuring that the models complemented each other to provide accurate and relevant recommendations.

Final Report Writing: Both team members collaborated on writing the final report, including the introduction, methodology, results, and discussion sections. They ensured that the report was comprehensive and accurately reflected the project's scope and outcomes.

Application Deployment: Sahar and Mahak jointly worked on deploying the application, ensuring that it was accessible to users and performed well under real-world conditions.

In summary, Sahar and Mahak contributed equally and significantly to the development of the personalised news recommender system. Their combined efforts in model training, backend and frontend development, integration, testing, and documentation resulted in a robust and user-friendly application that addresses the challenge of navigating vast amounts of online news content