News Article Text Classification Using Deep Learning

Deepakshi Mathur
M.S. Data Science, Arizona State University
GitHub: https://github.com/deepakshimathur
Email: dmathur4@asu.edu

Abstract

This project presents a comprehensive pipeline for classifying news articles into five distinct categories: **Politics**, **Sports**, **Business**, **Entertainment**, and **Technology**. The classification model was developed using various Natural Language Processing (NLP) techniques, advanced feature engineering methods (CountVectorizer, TF-IDF, GloVe, and BERT), and deep learning via neural networks. The best-performing model achieved over **97.5% accuracy** through rigorous cross-validation, demonstrating strong generalization capabilities.

---

1. Introduction

Text classification is a foundational task in Natural Language Processing (NLP). The objective of this project is to classify news articles into one of five predefined categories based on their content. Given the diversity and complexity of natural language, this task requires robust preprocessing, effective feature extraction, and a well-tuned classification model.

---

2. Dataset

- **Training Set**: 1063 news articles with labeled categories.
- **Test Set**: 735 unlabeled articles (predictions required).
- Each document consists of a unique article ID, free-form text, and (for training) a category label.

---

3. Text Preprocessing Pipeline

To prepare the raw text for modeling, a multi-stage preprocessing pipeline was developed:

- **Stemming** using NLTK’s Porter Stemmer
- **Punctuation removal** and **lowercasing**
- **Tokenization** into individual words
- **Stopword removal** using NLTK stopword list
- **Bigram generation** to capture contextual patterns
- **Keyword extraction** using YAKE (Yet Another Keyword Extractor)

This process ensured a clean and normalized text input for downstream feature extraction.

---

4. Feature Engineering

Multiple vectorization and embedding methods were tested to convert textual data into machine-readable features:

4.1 CountVectorizer

- Converts text into sparse matrices based on word frequency.
- Basic Bag-of-Words representation.

4.2 TF-IDF Vectorizer

- Reflects term importance by reducing the weight of commonly occurring words across documents.
- Limited to 10,000 most important features.

4.3 GloVe Embeddings

- 50-dimensional pre-trained word vectors from `glove.6B.50d.txt`
- Document embeddings created by averaging word vectors.

4.4 BERT Embeddings

- Contextual token embeddings via Hugging Face Transformers.
- Tokenization with BERT tokenizer and sentence encoding with `bert-base-uncased`.

---

5. Model Architecture

A **Multi-Layer Perceptron (MLP)** was trained using TensorFlow/Keras:

- **Input Layer**: Dimensionality varies based on feature type
- **Hidden Layers**: Two dense layers with 128 neurons each, ReLU activation
- **Output Layer**: Softmax activation for 5-category classification
- **Optimizer**: Adam, RMSprop, or SGD
- **Loss Function**: Sparse Categorical Cross-Entropy
- **Cross-validation**: 5-fold StratifiedKFold

---

6. Hyperparameter Tuning

- **Learning Rate Sweep**:
  Explored `[0.0001, 0.0003, 0.001, 0.003, 0.01, 0.03, 0.1]`
  Optimal: `0.001`
- **Optimizer Comparison**:
  - SGD
  - RMSprop
  - **Adam** (best validation performance)

---

7. Evaluation Results

Feature Method	Validation Accuracy	Final Training Accuracy
CountVectorizer	92.30%	98.40%
TF-IDF	95.62%	99.20%
GloVe	96.75%	99.80%
BERT	**97.55%**	**99.91%**

Final predictions were generated on the test set using the best-performing model (BERT + Adam).

---

8. Tools & Libraries

- **Languages**: Python 3.10+
- **Libraries**: TensorFlow, Keras, Scikit-learn, NLTK, Gensim, Transformers, YAKE
- **Notebook Environment**: Jupyter

---

9. Conclusion

This project demonstrates how combining preprocessing, embedding strategies, and neural networks can achieve high-accuracy results in text classification. BERT embeddings significantly outperformed traditional feature methods, showcasing the power of contextualized representations in modern NLP tasks.