Final Project Report

Introduction:

Text classification can be challenging nowadays with the big amount of data on the internet. Classification has become an important task to achieve. We aim to apply 2 Machine Learning approaches (Random Forest and SVM) and 2 Deep Learning approaches (LSTM and BERT) to see their performance classifying documents and comparing their results.

Methodology:

Our data is textual data containing text documents about tv news.

The dataset comprises 2,225 instances and 2 attributes: 'labels' and 'data.' The 'data' column contains the actual textual content from TV news, while the 'labels' column provides classifications for each text entry. There are five different classifications: Sport, Business, Politics, Tech, and Entertainment. The distribution of instances across these classes is as follows: Sport (511), Business (510), Politics (417), Tech (401), and Entertainment (386). Initially, both columns are of type object.

Our vocabulary consists of 64151 unique characters. From the below plot we can see the 50 most common characters in our dataset.

A graph of a number of blue bars

Description automatically generated

To process our textual data, we focus on several key preprocessing steps. First, we remove punctuation and special characters to eliminate any irrelevant symbols. We then apply tokenization and lemmatization to refine the text. For vectorization, we utilize Term Frequency-Inverse Document

Frequency (TF-IDF). Additionally, we employ label encoding to convert the categorical data into

numerical values, assigning numbers between 1 and 5 to represent the different labels.

Given that we only have 2 attributes we have decided to keep them and use them in our project.

Data mining task:

This is a classification task, where we aim to categorize documents based on their labels. Our goal is to develop a model that outperforms others in terms of accuracy and effectiveness.

Experiments and Results:

We have experimented with two machine learning algorithms and two deep learning approaches. Specifically, we used Random Forest and Support Vector Machine (SVM) for the machine learning algorithms. For our deep learning algorithms, we used the DistilBERT model, which is based on BERT and LSTM.

**Random Forest:**

The Random Forest was trained using n\_estimators = 100. We obtained the following results for each category:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-score |
| Business | 0.93 | 0.96 | 0.95 |
| Sport | 0.98 | 0.99 | 0.98 |
| Politics | 0.94 | 0.97 | 0.96 |
| Tech | 0.97 | 0.94 | 0.96 |
| Entertainment | 1.00 | 0.95 | 0.98 |

**Support Vector Machine:**

The SVM was trained using kernel='linear'. We obtained the following results for each category:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-score |
| Business | 0.97 | 0.94 | 0.95 |
| Sport | 0.99 | 1.00 | 0.99 |
| Politics | 0.96 | 0.97 | 0.97 |
| Tech | 0.96 | 0.97 | 0.97 |
| Entertainment | 0.96 | 0.97 | 0.97 |

**Long Short-Term Memory:**

The LSTM model has the following parameters: hidden\_size =256, num\_layers=2, dropout\_rate= 0.5. The hyperparameters used in the model are learning\_rate=0.01 and num\_epochs=10. We obtained the following results for each category:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-score |
| Business | 0.97 | 0.94 | 0.95 |
| Sport | 0.99 | 1.00 | 0.99 |
| Politics | 0.96 | 0.97 | 0.97 |
| Tech | 0.96 | 0.97 | 0.97 |
| Entertainment | 0.96 | 0.97 | 0.97 |

**BERT:**

The BERT model has the following parameters: num\_labels=5 (representing the number of classes in the classification task). The hyperparameters used for training include the learning rate (lr=5e-5), the number of epochs (3), and the use of a linear learning rate scheduler with no warmup steps. The loss function used is CrossEntropyLoss for classification tasks. We obtained the following results for each category:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Precision | Recall | F1-score |
| Business | 0.96 | 0.96 | 0.96 |
| Sport | 1.00 | 0.99 | 0.99 |
| Politics | 0.96 | 0.97 | 0.97 |
| Tech | 0.96 | 0.99 | 0.98 |
| Entertainment | 1.00 | 0.98 | 0.99 |

**Overall Performance:**

|  |  |  |
| --- | --- | --- |
| Model | Accuracy | Running Time (seconds) |
| SVM | 0.965 | 4.81 |
| Random Forest | 0.964 | 1.21 |
| LSTM | 0.968 | 16.53 |
| BERT | 0.985 | 503.94 |

Conclusions:

From our experiment and result section we have seen that our four models work well on our dataset, but BERT model works much better that LSTM, Random Forest, and SVM. Although BERT (Bidirectional Encoder Representations from Transformers) works best in our dataset, it can be challenging to implement and train because it is based on new techniques like transformers.

For future work, experimenting with different parameters and hyperparameters to optimize model performance, as well as exploring alternative algorithms, will be crucial for improving results.

References:

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150.

Gasparetto, A., Marcuzzo, M., Zangari, A., & Albarelli, A. (2022). A survey on text classification algorithms: From text to predictions. *Information*, *13*(2), 83.

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification?. In *Chinese computational linguistics: 18th China national conference, CCL 2019, Kunming, China, October 18–20, 2019, proceedings 18* (pp. 194-206). Springer International Publishing.

Song, G., Ye, Y., Du, X., Huang, X., & Bie, S. (2014). Short text classification: a survey. *Journal of multimedia*, *9*(5).

**Team 3:**

Ricardo Manjarrez

Carlos Torres

Christopher Biekeu

Mingfang Zhu