Fake News Stance Detection

Project Overview

The spread of misinformation on social media is a significant problem, addressed by the Fake News Challenge (FNC-1) through the use of artificial intelligence, including machine learning and natural language processing. A key part of this challenge is Stance Detection, which examines how the body of a news article aligns with its headline. This technique checks if the article supports, contradicts, or is unrelated to the headline, aiding in the identification of potentially false news.

Our project delves into this problem by leveraging the FNC-1 dataset, exploring several classification models to discern the stance of news articles. These models fall under three primary approaches:

1. **Deep Learning** – Employing novel neural network architectures.
2. **Feature Extraction (Tree-Based Approach)** – Using decision trees and ensemble methods after feature engineering.
3. **Hybrid Architecture** – Combining the strengths of deep learning with feature extraction techniques.

Our aim is to maximize the fnc-1 score and class-wise F-1 scores while optimizing for training efficiency. This report outlines our exploration into these approaches and presents a system that achieves competitive performance with state-of-the-art architectures, with further research needed to validate its efficacy under varying conditions.

Stance Detection and Its Implications

Stance Detection is a nuanced task that involves classifying the relationship between a headline and the body text of news articles into four categories:

- **Agrees:** The body text is in agreement with the headline.
- **Disagrees:** The body text is in disagreement with the headline.
- **Discusses:** The body text discusses the same topic without taking a stance.
- **Unrelated:** The body text is on a different topic than the headline.

Understanding and accurately classifying these relationships is crucial for automating the identification of fake news.

Methodology

Our methodology is centered around classifying the stance of the body text relative to the headline into the aforementioned categories, employing three distinct approaches, each elaborated within the Models Section.

Dataset Description and Scoring Metrics

The dataset consists of 1,648 distinct headlines and 1,683 distinct articles, contributing to a total of 49,972 headline-article pairings. Given the disproportionate bias towards unrelated pairs within the dataset, the

FNC-1 challenge introduced a weighted accuracy score to supplement traditional F-1 metrics, which we also adopt to evaluate our models.

Folder Structure

Here's how the project repository is organized:

- **01_dataset/**: Contains the preprocessed datasets used in this project, including training, validation, and test sets. Subfolder also contains original dataset.
- **02_preprocessing/**: Scripts and notebooks for data cleaning and preparation. This includes text normalization, tokenization, and vectorization.
- **03_models/**: Jupyter notebooks for each model used in the approaches described below.
- **04_gui/**: Contains the web gui where users can upload their input csv file and get a csv export of the predictions.

Data Fields

- `Headline` - The headline of the news article.
- `Body_ID` - A unique identifier for the body text.
- `Stance` - The stance of the body text relative to the headline, categorized as `Agree`, `Disagree`, `Discuss`, or `Unrelated`.
- `ArticleBody` - The body text of the article.

Models

Approaches, Models and Associated Files

Approach 1: Deep Learning

We utilize different architectures to process text data through neural networks that leverage GloVe word embeddings.

1. LSTM Model Variants (`glove-bi-lstm-notst.ipynb`)

- **LSTM**
- **Bi-LSTM**
- **BiLSTM with Batch Normalization**

2. GloVe Embedding Models (`glove-with-gru-bigru-cnn-mp.ipynb`)

- **LSTM**
- **Bi-LSTM**
- **BiLSTM with Batch Normalization**
- **GRU**
- **GRU with Batch Normalization**
- **BiGRU with Batch Normalization**
- **CNN with Batch Normalization**
- **CNN with Max Pooling and Batch Normalization**

3. Baseline Model (`lstm-one-hot-2.ipynb`)

- **One Hot Encoded LSTM**

Approach 2: Feature Extraction (Tree-Based Models)

1. AI_RandomForest (`AI_RandomForest.ipynb`)

- **TF-IDF + Cosine Similarity**
- Model Variants:
  - **Random Forest**
  - **Multinomial Naive Bayes**
  - **Gradient Boosting**
  - **Linear SVM**
  - **Decision Tree**
  - **Logistic Regression**

2. XGBoost Models

- **Standard XGBoost** (`Xgboost-model`)
- **XGBoost Stacked with Naïve Bayes** (`XGBoost-model-stack-naïve-bayes`)

3. LightGBM Models

- **LightGBM Stacked with Naïve Bayes** (`LightGBM-model-stack-naïve-bayes`)

Approach 3: Hybrid Model (Combination of Approaches 1 and 2)

1. Hybrid CNN + Bi-LSTM (`hybrid_cnn_bilstm_concat.ipynb`)

- **CNN and Bi-LSTM Hybrid**

2. Hybrid BiLSTM and GRU (`hybrid_concat_bilstm_gru.ipynb`)

- **Concatenated Word Embedding ⇒ BiLSTM ⇒ GRU**
- **TF-IDF Vectoriser**
- **MLP**

3. Stacking Models with Polarity Features (`Stacking-polarity.ipynb`)

- **TF-IDF with WordNet**
- **MLP**

4. Improved TF-IDF Vectoriser (`tf-idf-vectoriser-bug-removed.ipynb`)

- **MLP**
- **LSTM**

- **BiLSTM**
- **BiLSTM with Max Pooling**

Final Best Model

MLP Best Model (`MLP_best_model.ipynb`)

- **Multilayer Perceptron**
- **Best Model Checkpoint**

Installation

Clone the repository:

```
git clone https://github.com/joebaarath/ai_project2024.git


### Setting Up  Web App GUI

## Web Flask GUI Setup

Follow these steps to set up the Flask web application for stance detection:

1) Create a new virtual environment:
python -m venv venv

2) Activate the environment:
python/web/host

3) Install the required packages:
python -m venv venv

4) Upload fake news csv (i.e. lemmatized_dataset_final_balanced_test.csv )

5) Run the Flask web application:
```