Fake News Detection Using NLP

PROPOSED SOLUTION

As we have seen that the problem of spreading fake news is a serious issue therefore, there is a need to detect this fake news. The main aim of the project is to obtain a model which will help in detecting if a news article is fake or not. The problem of detecting fake news is a very difficult task and many researchers are trying to obtain a solution to it.

Since there are not many datasets which are available publicly to perform this task. We have considered three different datasets which will be merged together to obtain a master dataset which will help in training the models to find if a news is fake or not. Firstly, the datasets are collected. The datasets are then merged to obtain a master dataset. This dataset is then preprocessed. Preprocessing of the datasets include lowering of the data, stop word removal, stemming, tokenization and padding is also performed in order to obtain the same length. The dataset is then split into training data and testing data.

To overcome the problem of detecting fake news this project proposes 6 similar LSTM models which are to be trained and each model will be fed with the different text vectors of news headline and news content. This will help in obtaining a good model which will tell if the news is true or it is fake. In this project we have used six similar LSTM models.

Three text vectorization techniques are used which are GloVe, Word2vec and TF-IDF. The first LSTM model will be fed with the vectors of the title of the news using GloVe. The second model will be fed with the vectors of the content of the news using GloVe. Similarly, two models will be built using the Word2vec technique each for the title of the news and the content of the news respectively. Lastly, the LSTM model will be fed with the text vectors of the title of the news using TF-IDF and another model will be fed with the text vectors of the content of the news using TF-IDF. By doing so we can identify which technique gives better results and identify which model performs well. Lastly, the performance is measured using the performance metrics accuracy, precision and recall.

A. Dataset

There are very few datasets which are available publicly for the detection of fake news. In this paper we have used three different datasets which are available online. The first dataset ISOT Fake News dataset is obtained from a website [17]. The second data that is used in the project is the Fake News Detection dataset from Kaggle [18]. The third dataset used is the Real and Fake News Dataset which is obtained from Kaggle [19].

B. Merging the Dataset

The first dataset ISOT Fake News dataset is obtained from a website [9]. This dataset was created using data from real world news sources. This dataset consists of two types of articles: fake and real. The dataset consists of two CSV files. First file contains all the news which is true and the second file contains the news which is fake.

Each article contains the following information: article title, text, type and the date the article was published on. The second dataset used in the project is the Fake News Detection dataset from Kaggle. This dataset consists of 4 columns which are the URLs of the news source, the Headline of the news, the Body of the news that is the content of the news and the last column contains the Label of the news which tells whether the news is fake or not. Next, the two datasets are merged together to obtain a single dataset. After the merge we obtained a dataset with 10344 records. Finally, we obtain a master dataset by merging the first dataset with the above merged dataset [dataset with 10344 records], hence the final obtained master dataset consists of 54726 records and three columns, Title, text and Class.

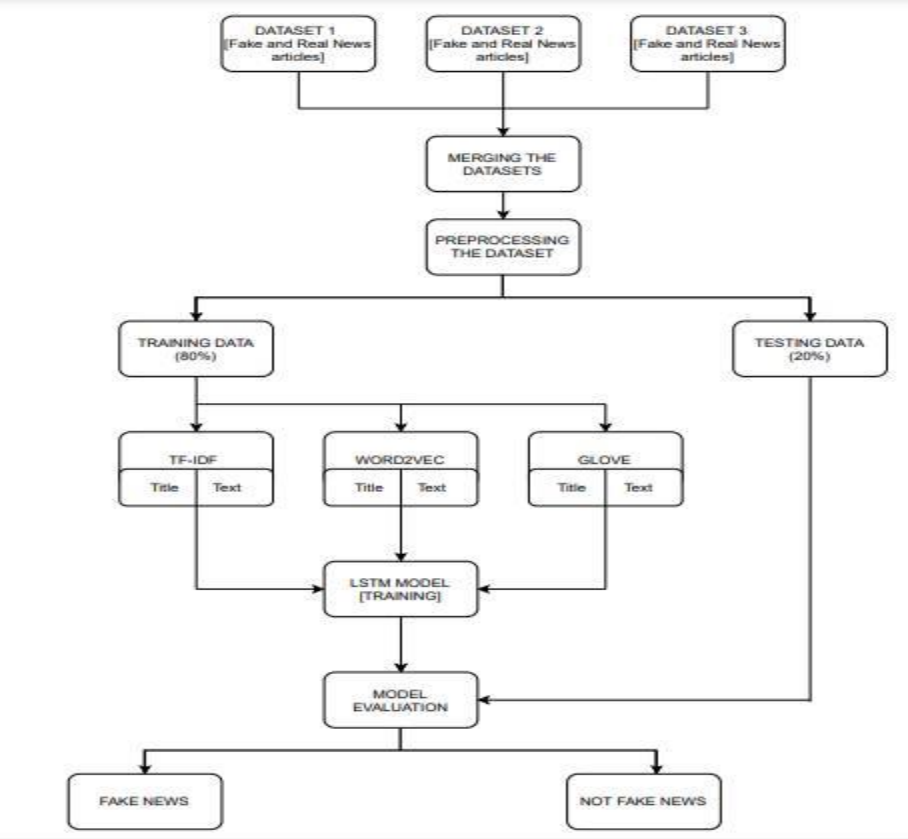

Fig.1 Proposed System Model for Detection of Fake News.

C. Preprocessing the dataset

Data preprocessing is a data mining technique that involves transforming raw data into understandable form. In natural language processing, text preprocessing is the practice of cleaning and preparing text data. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing methods such as tokenization, lemmatization, stop word removal and lowercasing.

D. Train -Test Split

The next step in the process is to split the data into train and test data. Here, we have done 80% train data and 20% test data split.

E. Feature Extraction

The next step is feature extraction. Machine Learning algorithms learn from a predefined set of features from the training data to produce output for the test data. But the main problem in working with language processing is that machine learning algorithms cannot work on the raw text directly. So, we need some feature extraction techniques to convert text into a matrix (or vector) of features.

Three feature extraction methods will be used

1) TF-IDF: TF-IDF stands for term frequency-inverse document frequency. It highlights a specific issue which might not be too frequent but holds great importance. The TF–IDF value increases proportionally to the number of times a word appears in the document and decreases with the number of documents in the corpus that contain the

word. TF-IDF(Term Frequency/Inverse Document Frequency) is one of the most popular IR(Information Retrieval) techniques to analyze how important a word is in a document. TF-IDF is the product of TF and IDF. A high TF-IDF score is obtained by a term that has a high frequency in a document, and low document frequency in the corpus. For a word that appears in almost all documents the IDF value approaches 0, making the tfidf also come closer to 0. TF-IDF value is high when both IDF and TF values are high i.e. the word is rare in the whole document but frequent in a document.

2) WORD2VEC: Word2Vec produces a vector space, typically of several hundred dimensions, with each unique word in the corpus such that words that share common contexts in the corpus are located close to one another in the space. That can be done using 2 different approaches: starting from a single word to predict its context (Skip-gram) or starting from the context to predict a word (Continuous Bag-of-Words). Word2vec is one of the most popular implementations of word embedding, which was invented by Google in 2013. It describes word embedding with two-layer shallow neural networks in order to recognize context meanings. Word2vec is good at grouping similar words and making highly accurate guesses about meaning of words based on contexts. It has two different algorithms inside: CBoW (Continuous Bag-of-Words) and skip gram model.

3) GLOVE: Glove, a very powerful word vector learning technique Glove does not rely just on local statistics (local context information of words), GloVe (Global Vectors for Word Representation) is an alternate method to create word embedding's. It is based on matrix factorization techniques on the word-context matrix. A large matrix of co-occurrence information is constructed and you count each "word" (the rows), and how frequently we see this word in some "context" (the columns) in a large corpus.

F. Model

The model that will be used in this project is the LSTM model. The features extracted from the above feature extraction methods will be given to the LSTM model.

All the pre-processed news titles and content in vector form are given to the LSTM model. We have used the Tensorflow framework to perform this task of detecting fake news. Long Short Term Memory [LSTM] Long short-term memory networks are an extension for recurrent neural networks, which basically extends the memory. The units of an LSTM are used as building units for the layers of a RNN, often called an LSTM network.

LSTMs enable RNNs to remember inputs over a long period of time. This is because LSTMs contain information in a memory, much like the memory of a computer. The LSTM can read, write and delete information from its memory. This memory can be seen as a gated cell, with gated meaning the cell decides whether or not to store or delete information (i.e., if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm.

This simply means that it learns over time what information is important and what is not. In an LSTM you have three gates: input, forget and output gate. These gates determine whether or not to let new input in (input gate), delete the information because it isn't important (forget gate), or let it impact the output at the current timestep (output gate).