A Deep Learning Approach to Fake News Detection

1. References to at least two scientific papers that are related to your topic

Shu, K., Sliva, A., Wang, S., Tang, J. and Liu, H., 2017. Fake news detection on social media: A data mining perspective. *ACM SIGKDD explorations newsletter, 19*(1), pp.22-36.

Farokhian, M., Rafe, V. and Veisi, H., 2022. Fake news detection using parallel BERT deep neural networks. *arXiv preprint arXiv:2204.04793*.

2. A decision of a topic of your choice

A Deep Learning Approach to Fake News Detection:This research focuses on developing a neural network model (CNN or RNN) to classify news articles as legitimate or fake. The idea is to use deep learning algorithms to detect patterns in news article text that can distinguish between real and fake content. The project's goal is to investigate the potential of neural networks in automating the detection of false news by evaluating massive datasets of labeled articles.

3. A decision of which type of project you want to do

Bring your own method: This project will use an existing neural network architecture (CNN or RNN) to detect fake news. I will use Kaggle's Fake and Real News Dataset, which includes tagged news items. The project's main goal will be to train a model that takes advantage of either CNN's capacity to find patterns in n-grams or RNN's ability to capture sequential dependencies between words. The goal is to train and fine-tune a strong text categorization model for outstanding performance.

4. A written summary that should contain:

a. Short description of your project idea and the approach you intend to use

In this project, I will create a neural network (either CNN or RNN) to determine whether news articles are authentic or fraudulent. The CNN model will be intended to detect n-grams and word patterns that are suggestive of bogus news, whereas the RNN model will learn the text's sequential structure. Tokenizing the text, deleting stop words and punctuation, and transforming it to vector representations using TF-IDF or Word2Vec embeddings are all part of the preprocessing procedures. The model will then be trained on the preprocessed data, and its performance will be measured using accuracy, precision, recall, and F1-score. A CNN focuses on detecting relevant n-grams, whereas an RNN captures long-range relationships between words.

b. Description of the dataset you are about to use (or collect)

The dataset to be used is the **Fake and Real News Dataset** from Kaggle, which consists of thousands of labeled news articles. These articles are labeled as either **real** or **fake**, and the textual content covers a wide variety of topics and news sources. The dataset will be preprocessed by tokenizing the text, removing stop words, punctuation, and then converting the text into numerical embeddings using techniques like **TF-IDF** or **Word2Vec**.

Key Dataset Features:

- **News Articles**: Thousands of labeled articles (real or fake).
- **Textual Features**: Cleaned and tokenized text, converted to vector embeddings (TF-IDF, Word2Vec).

Preprocessing Steps:

- Remove stop words, punctuation, and special characters.
- Tokenize the text and convert it to vector embeddings.

c. A work-breakdown structure for the individual tasks with time estimates (hours or days) for dataset collection, designing and building an appropriate network, training and fine-tuning that network, building an application to present the results, writing the final report, and preparing the presentation of your work.

Oct. 22nd - Start

Oct. 25th - Dataset preparation and cleaning (3 days)

- Load the dataset and clean the text data (remove stop words, punctuation, etc.)
- Perform tokenization and convert the text into numerical representations (e.g., TF-IDF, Word2Vec)

Oct. 28th - Model design(3 days)

- Build the deep learning model (CNN or RNN)
- Define the loss function and evaluation metrics (binary cross-entropy, accuracy, precision)

Nov. 4th - Model training (1 week)

- Train the model on the training set
- Monitor performance using metrics like accuracy and loss

Nov. 11th - Evaluation and tuning (1 week)

- Evaluate the model on the test set and calculate performance metrics (accuracy, precision, recall, F1-score)
- Fine-tune the model by adjusting hyperparameters such as learning rate and batch size

Nov. 18th - Finalization and visualization (2 weeks)

- Visualize performance with confusion matrix and accuracy/loss plots

- Analyze misclassified articles and make improvements to the model

Dec. 2nd - Write report and prepare presentation (4 weeks)

- Summarize findings, including performance metrics and recommendations for improvement
- Create visualizations for the presentation