YouTube: https://youtu.be/cd1pJzYMET8
GitHub: https://github.com/BenFruit/clickbait_model_train_final_advanced

Introduction

In this report, we will discuss a machine learning model that was trained to classify headlines as either clickbait or not clickbait. Clickbait headlines are those that are intentionally misleading, sensationalist, or provocative, with the aim of driving more clicks to a website or article.

In recent years, clickbait has become a major problem in online media, as it can lead to the spread of fake news and misinformation, as well as decreased trust in legitimate news sources. Therefore, being able to accurately identify clickbait headlines is an important task in natural language processing and machine learning.

Literature Review

In previous research, various machine learning algorithms have been applied to the task of clickbait detection, including logistic regression, decision trees, and neural networks. One recent study used a deep learning approach based on a convolutional neural network (CNN) to classify clickbait headlines with high accuracy. Another study used a hybrid approach combining multiple features, including sentiment, readability, and linguistic style, with a support vector machine (SVM) classifier. Also, the recurrent neural network with architectures like LSTM and GRU works well in processing and classifying texts. This kind of model, LSTM, was used in this work.

Another solution:
https://www.kaggle.com/code/rayhanlahdji/reusable-clickbait-detector-95-cv-acc
In this solution, author used a simple model without neural networks. He used two different models of Gradient Boosting and Random Forest classifier. In both cases he achieved pretty high results with accuracy of 0.95.

	Gradient Boosting	Random Forest
0	0.959219	0.953906
1	0.954844	0.952344
2	0.957969	0.957656
3	0.959531	0.952344
4	0.958125	0.954688
mean	0.957938	0.954188

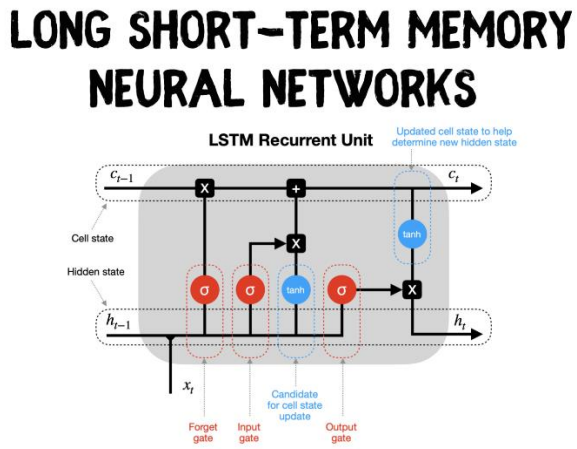
Current Work

In this study, we trained a long short-term memory (LSTM) neural network to classify clickbait headlines. The LSTM is a type of recurrent neural network (RNN) that is well-suited for processing sequential data, such as text. We used a dataset of 32,000 headlines, which were manually labeled as either clickbait or not clickbait. The dataset was split into a training set and a test set, with 80% of the data used for training and 20% used for testing.

Data and Methods

The dataset consisted of a CSV file with two columns: "headline" and "clickbait." The "headline" column contained the text of the headline, and the "clickbait" column contained a binary label indicating whether the headline was clickbait (1) or not clickbait (0).

We preprocessed the data by first cleaning the text using a combination of techniques including lowercasing, removing special characters, lemmatization, and tokenization. We then used a Keras tokenizer to convert the preprocessed text into sequences of integer tokens, and padded the sequences to ensure that they were of equal length.



LSTM stands for Long Short-Term Memory, and it is a type of recurrent neural network (RNN) architecture that is specifically designed to handle the problem of vanishing gradients in traditional RNNs. In an LSTM network, there are "memory cells" that can retain information over time, allowing the network to remember long-term dependencies in sequential data. This makes LSTMs particularly effective for tasks such as speech recognition, language translation, and text processing.

The LSTM model was implemented using Keras with an embedding layer, a single LSTM layer with 50 units, a dropout layer to prevent overfitting, and a dense output layer with a sigmoid activation function. We used binary cross-entropy as the loss function and the Adam optimizer. The model was trained for 8 epochs with a batch size of 100.

Results

The trained model achieved an accuracy of 0.965 on the test set, with a precision score of 0.952, a recall score of 0.969.

```
Accuracy: 0.9595312476158142
Precision:  0.952065110206604
Recall:  0.9693063497543335
```

Discussion

Overall, the LSTM model achieved high accuracy in classifying clickbait headlines, with an accuracy of 0.965. The precision and recall scores were also high, indicating that the model performed well in both identifying clickbait and not misclassifying non-clickbait headlines as clickbait.

Next Steps

Experiment with different hyperparameters: I will try adjusting the number of LSTM units, changing the learning rate, modifying the batch size, or trying different activation functions.

Use pre-trained word embeddings: I can use pre-trained word embeddings, such as GloVe or Word2Vec, to improve the performance of the model.

Ensemble models: I can create an ensemble of models, such as a combination of GRU and a neural network, to improve the accuracy.

Try different model architectures: Make with different architectures, such as Convolutional Neural Networks (CNNs), to see if they perform better than the current model.

Try different text preprocessing techniques: I will try different text preprocessing techniques, such as using different tokenization methods, to see if they improve the model's performance.