Disinformation and misinformation are pervasive in western societies, causing significant rifts and partisanship between different socioeconomic groups and races. Much of the increasing partisanship has been attributed to disinformation and misinformation spread on social media platforms such as Facebook and Twitter. Recently Facebook and Twitter have developed models for identifying fake news articles and flagging those for readers to be aware of. However, clickbait headlines and articles aren't identified by this. Similar disinformation and misinformation can be spread through overly grandiose headlines and misleading images.

In addition to spreading misinformation and disinformation, clickbait articles can have significant mental health impacts on social media and internet users. Many articles will portray "idealistic" archetypes of societal characters, such as wives, mothers, fathers, teens, etc. These idealized versions of people can distort people's self images, and create misleading and unrealistic ideas of who one should be in society. These distorted images and archetypes can lead to significant mental health degradation by promoting body dysmorphia, social isolation and anxiety, and other insecurities. In order to help curb these misleading headlines and images I built a Long Short-Term Memory (LSTM) model to identify clickbait headlines from an existing data set with 95% accuracy.

For my model I used two critical natural language processing and deep learning functions. The first was Word2Vec from the python package Gensim. Word2Vec, as the package name describes, vectorizes words, or converts words into numeric arrays. This helps computers better process and "intuit" novel words similar to how we can. There are two main methods for vectorizing words (Figure 1). The first method, CBOW, uses surrounding words to predict a current word. The second method, skip-gram, does the opposite, and predicts surrounding words based on the current word. I employed a CBOW model for my project. LSTM models function as more complex recurrent neural network models (RNNs). RNNs are extremely useful in processing strings of data such as text, audio, and video files, but the standard RNNs suffer from data degradation. In text data this is caused by having less important words weighted the same as important words, and when you have long enough text files, those weights all trend towards 0. LSTM rectifies this by adding a Forget Gate to the model and some other transformations within the nodes (Figure 2).

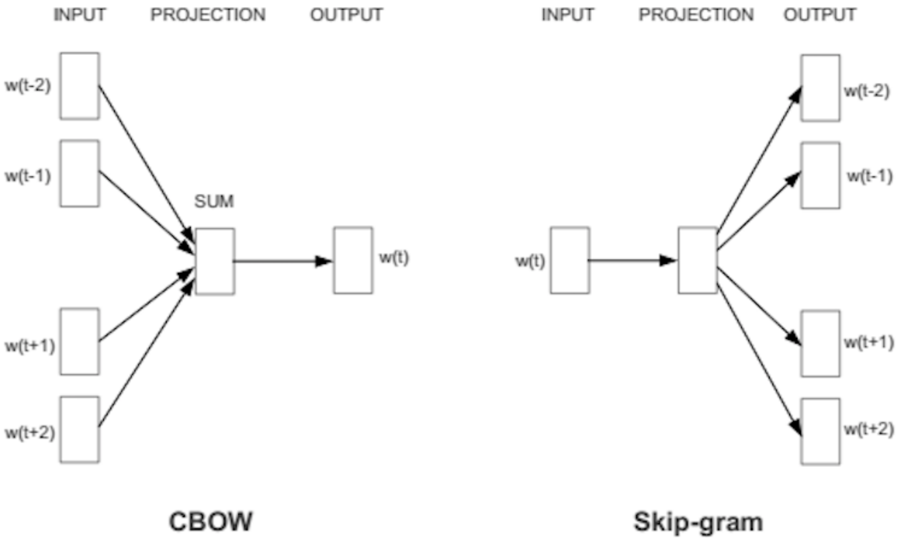

Figure 1: Illustration of Word2Vec function from Wiki Commons (2021).

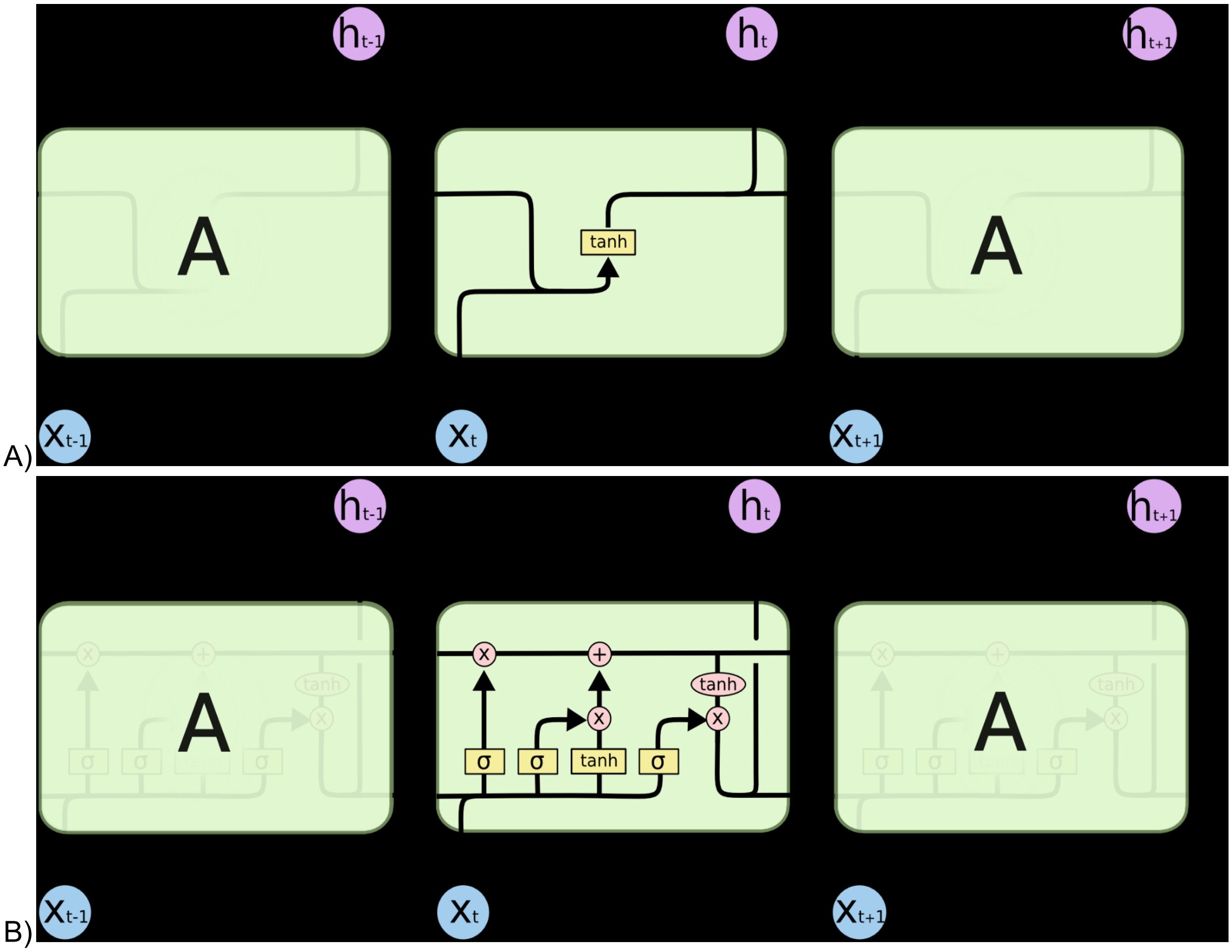

Figure 2: Illustration of simple recurrent neural network (A). Illustration of Long-Short Term Memory (B).

The dataset that I worked with for this project is available at Kaggle.com, and contains 32,000 clickbait headlines classified as either clickbait or not clickbait. Before building my LSTM model, I wanted to inspect the data for any existing trends. First I compared the sizes of the two data sets, and found them to be almost exactly equal (15,999 clickbait headlines and 16,001 not clickbait headlines) (Figure 3). I then calculated the average length of headline for each category and found clickbait headlines to be longer on average than not clickbait headlines (9.94 words vs. 8.19 words) (Figure 4). Lastly, I created word clouds of each category to identify what words are most commonly used for each, these are depicted in Figures 5 and 6.

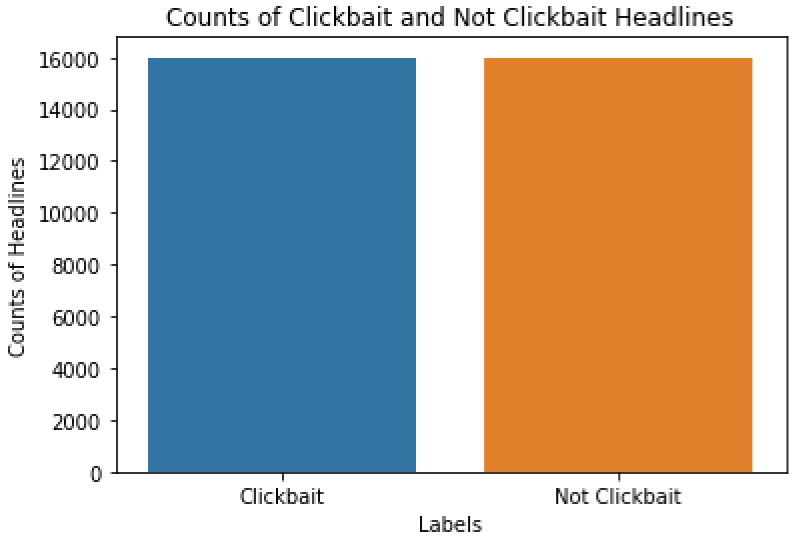

Figure 3: Data distributions of clickbait and not clickbait headlines.

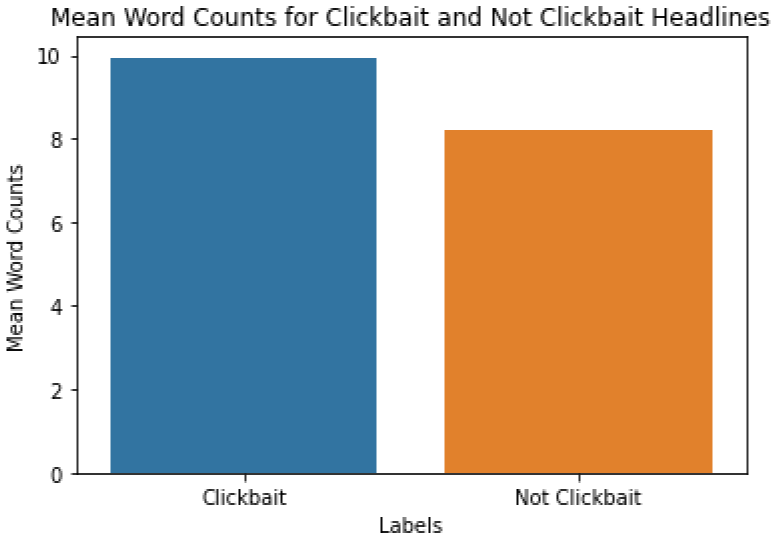

Figure 4: Mean word count for all clickbait and not clickbait headlines.

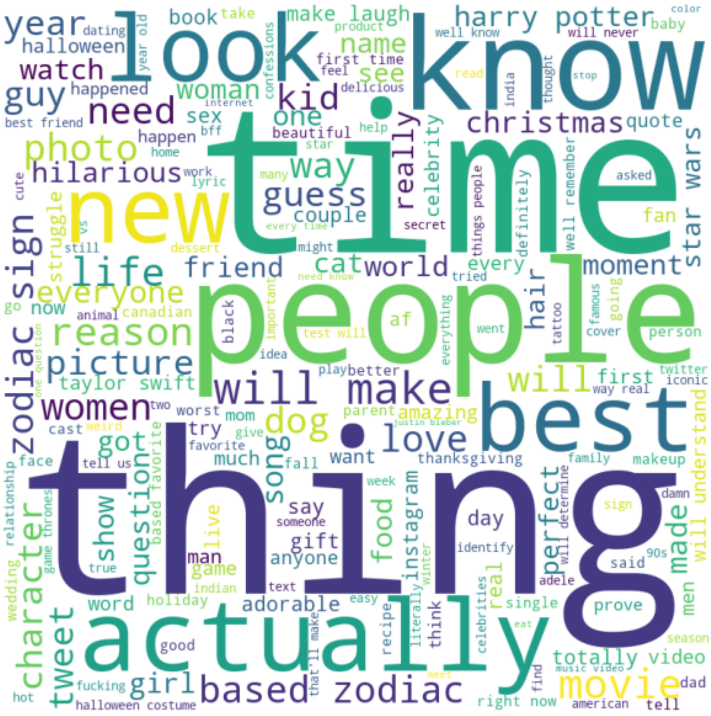

Figure 5: Word cloud of clickbait headlines.

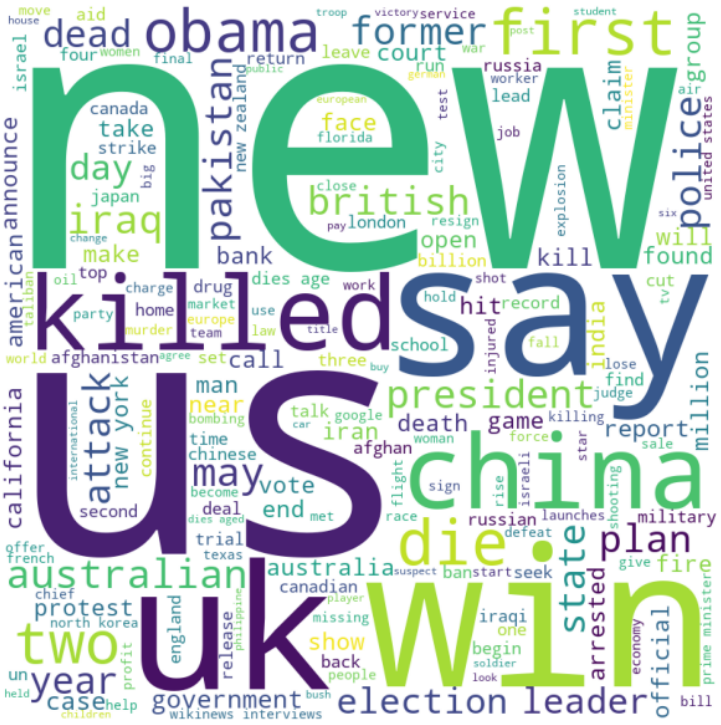

Figure 6: Word cloud of not clickbait headlines.

When building the models I vectorized the words, then split the data into an 80% training set and 20% testing set. Lastly, I built the LSTM model to fit the data over 3 epochs. I chose 3 epochs because when training on 20 epochs there was a consistent over fitting of data to the training set after the second or third epoch (Figure 7). After building the model and testing it, I found that it accurately identified clickbait headlines 98% of the time, with 98% precision and recall as well (Figure 8). This met and exceeded my goal of a model with 95% accuracy, precision and recall.

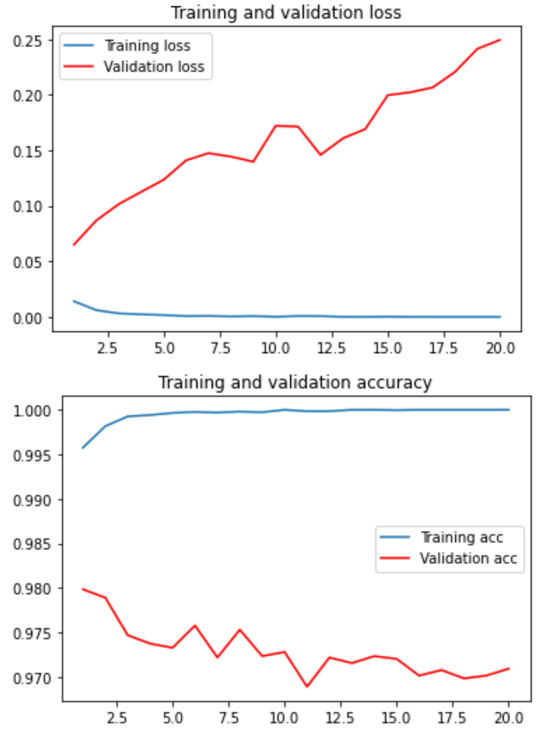

Figure 7: Accuracy and validation of training and testing data sets across 20 epochs.

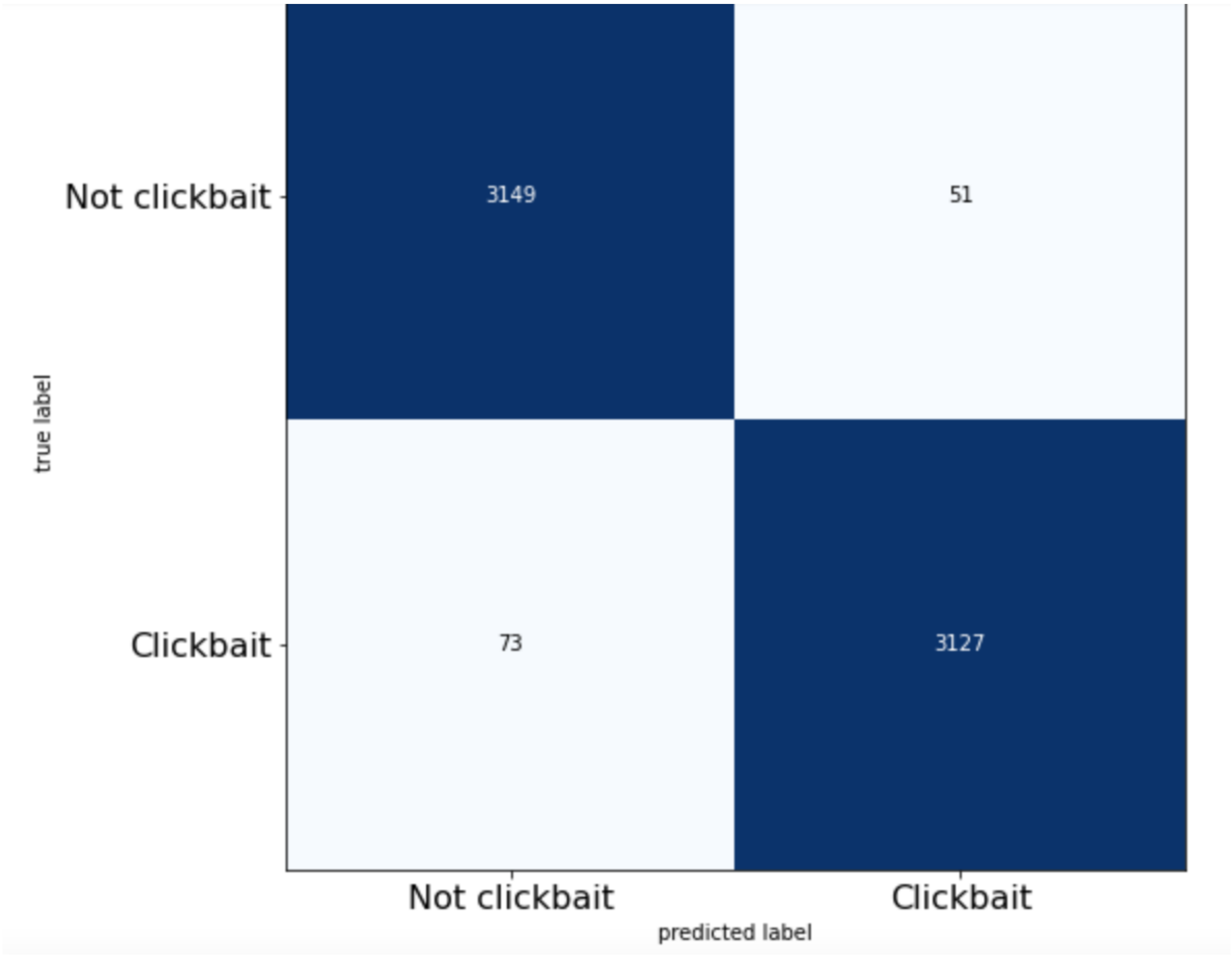

Figure 8: Confusion matrix of true and predicted labels from the model.

After succeeding in building the model with over 95% accuracy, my next goal would be to employ the model on actual headlines with links, and include some functions from the python package newspaper3k. This package allows a user to get the author of an article, publisher, publication date, and a brief summary of that article. Adding these functions to a well-performing clickbait identification model would help social media and internet users not only identify clickbait headlines, but also get a quick summary of what the article actually discusses and potentially identify consistent authors and publishers of clickbait articles.