COVID-19 Fake News Detection

By: Vincent DiPerna, Michael Blair, Sungjeon Choi, and Safa Kamran
Course Instructor: Mohammed Farag
CS 4624 Multimedia/Hypertext Capstone
Virginia Tech, Blacksburg, VA 24061
4/18/2023
Outline

- Background
- Categories
- Completed Work
- Preprocessing Script
- Preprocessing Script Output
- Preprocessing Script Statistics
- Categories Chosen
- AI Model Implementation (Preprocessing)
- Formatting Training Data
- AI Model Implementation (Training)
- AI Model (Testing)
- Website Implementation
- Backend
- Flask Server
- Added Functionality
- To-Do
- Challenges
- Lessons Learned
- Future Work
- Acknowledgements
- References
Background

- Problem: Misinformation about Covid-19 spread on Twitter
- Goal: Help identify/categorize tweets that are misinformation
- Solution: Web page where you can input a tweet and it will categorize the tweet
Completed Work

- Preprocessing
- Back-end
- Front-end
- AI Model
Preprocessing Script

- The script is invoked on the command line using “main.py” followed by the JSON file that is to be analyzed.
- The JSON file must contain a JSON list of JSON objects.
- The script first opens the files, and loads the AI models into memory.
- Each tweet is parsed, and added to a SQL commands file for later integration into a MySQL database.
- The text is further parsed and then running through the AI model.
The links from each tweet are also saved in a file.

The script can process about 1000 tweets per second, and the limiting factor is the model which takes a significant amount of time to run.

Multithreading was considered, but was not implemented in the end.

- The number of tweets was so large that dividing up the work for separate threads proved to be time consuming and very memory consumptive.
- The combination of the files afterward had a lot of overhead as well.

The script will output some statistics on the tweets analyzed including the percentage of tweets in each classification category.
Preprocessing Script Output

- `sqlcommands.txt` - A text file containing the INSERT SQL commands for inserting data into a SQL database
- `tweettexts.txt` - A file containing the text of each tweet nicely formatted for the AI model training
- `tweetlinks.txt` - A file containing all the links associated with each tweet that are not twitter links
Preprocessing Script Statistics

● The statistics from running the script on about 1 million tweets was
  ○ Vaccine Misinformation - 34.2%
  ○ Treatment Misinformation - 13.2%
  ○ Other (including Truth) - 52.9%

● Since there were are only two misinformation categories, the results are a skewed a bit towards vaccine misinformation

● With more categories and a wider range of test data, the model could have been made more accurate

● However with the limited amount of time for this project, we were unable to train the model with more categories
Categories Chosen

- Treatment
- Vaccine
- Other
AI Model Implementation (Preprocessing)

- 17.5 Million Tweets
- Manually filter tweets after keyword match

```python
# open text file and read lines with UTF-8 encoding
with open('tweettexts.txt', 'r', encoding='utf-8') as f:
    lines = f.readlines()

# create pandas DataFrame from lines
df = pd.DataFrame(lines, columns=['tweet'])

# drop duplicate tweets
df.drop_duplicates(subset='tweet', inplace=True)

# specify lists of vaccine-related and COVID-related keywords
vaccine_false_keywords = ['autism', 'microchip', 'infertility', 'population control', 'toxic', 'unsafe', 'experimental', 'track
vaccine_keywords = ['causes', 'immunization', 'vaccine', 'mms', 'mmy', 'antivax', 'covid vaccine', 'covid19 vaccine', 'covid 19

# filter tweets based on keywords
vaccine_df = df[(df['tweet'].str.contains(''.join(vaccine_false_keywords))) &
(df['tweet'].str.contains(''.join(vaccine_keywords)))]

# write filtered DataFrame to CSV with UTF-16 encoding
vaccine_df.to_csv('vaccine_tweets.csv', index=False, encoding='utf-8')```
Formatting Training Data
AI Model Implementation (Training)

- **Model:** Support Vector Classification (SVC)
- **Split:** 80/20

```python
# Impute missing values with an empty string
combined_df['tweet'] = combined_df['tweet'].fillna('')

# Split the dataframe into a training set and a testing set
train_df, test_df = train_test_split(combined_df, test_size=0.2, random_state=random_state_value)

# Convert the text data into a numerical representation using TF-IDF
tfidf = TfidfVectorizer()
X_train = tfidf.fit_transform(train_df['tweet'])
X_test = tfidf.transform(test_df['tweet'])

# Define the classification model
model = SVC(random_state=random_state_value)

# Train the model on the training set
model.fit(X_train, train_df['label'])
```
AI Model (Testing)

- Metrics from sklearn
- Confusion Matrix

Accuracy: 0.8602150537634409
Precision: 0.8626984126984126
Recall: 0.860215053763441
F1 Score: 0.8595246715557913
Confusion matrix:
```
[[59  0  3]
 [ 2 52  8]
 [ 9  4 49]]
```
Website Implementation

- MySQL Database
- Wildfly Server run locally
- JSF / XHTML frontend
- Java backend

- EC2 server run via AWS
- Flask server running AI model for detection
Backend

- Java backend
  - Two Controllers -> Links File Download, Tweet Controller
  - Tweet Entity Bean
  - Tweet Facade Bean
Flask Server

- Impossible to call python AI script via JSF
- Calling Flask server through Java backend
  - POST request URL set up
  - BufferedReader to get output
Added Functionality

- Downloadable text file of links connected to tweets
- PDF viewer of final report
- User input for AI detection
To-Do

● 4/27: VTechWorks final draft submission
  ○ Report
  ○ Presentation
  ○ Source code and remaining deliverables
● 5/2: Final checklist for VTechWorks and project peer reviews
  ○ Report
  ○ Dataset
  ○ Code
  ○ Software
Challenges

- Manually detecting tweets is tedious
- Spelling errors in tweets
- Massive Size of Files
- Inexperience with certain technologies
- Components dependent on each other
Lessons Learned

- Start work on milestones early
  - Task may take longer than expected
  - Questions arise when work begins
- Size constraints need to be addressed early in the project
- Communication is key
Future Work

- Making the preprocessing script multithreaded
- Creating more categories for the AI model and training it
- More data to train the AI model with
- Potential use for a better/more advanced model
- Deploy everything to EC2 for final presentation
Acknowledgements

- **Client/Instructor:** Dr. Mohamed Farag
  - Email: mmagdy@vt.edu
- **UTA:** Enmu Liu
- **Previous Team:** Eric Wiley, Tung Nguyen, Fareeza Zameer, Ferrin Kirby, Kyle Toroc, and Campbell Dalen

