A NOVEL SYSTEM FOR REGIONAL TWITTER HATE SPEECH ANALYSIS AND DETECTION USING DEEP LEARNING MODELS AND WEB SCRAPING

Nicole Ma[1], Yu Sun[2]

[1]Sage Hill School, 20402 Newport Coast Dr, Newport Coast, CA 92657
[2]California State Polytechnic University, Pomona, CA, 91768,
Irvine, CA 92620

ABSTRACT

Instances of hate speech on popular social media platforms such as Twitter are becoming increasingly common and intense. However, there still exists a lack of comprehensive deep-learning models to combat Twitter hate speech. In this project, a comprehensive detection and reporting platform, entitled "TweetWatch," was created to solve this issue. A binary classification CNN (Convolutional Neural Network) and a multi-class CNN were created to detect hate speech from real-time Twitter data and classify tweets with hate speech into five categories. The binary classification model has an AUC score of 98.95% and an F1 score of 97.88%. The multi-class classification model has an AUC score of 89.46%. All metrics reached over a targeted 5% increase from previous models in multiple papers, validating the proposed solution. Additionally, the only real-time choropleth map for hate speech in the United States was successfully created.

KEYWORDS

Web scraping, Natural language processing, Deep learning, Neural networks

1. INTRODUCTION

Online instances of hate speech are extremely common on virtually all social media platforms [1][2][3]. Based on previous research, 53% of Americans said they were targeted by hateful speech online and 37% reported severe attacks, but sites like Twitter still rely on an artificial intelligence algorithm that is only around 50% effective. This algorithm often misses instances of hate speech, which are usually targeted towards marginalized groups that already face so much turbulence in real life.

Deep learning methods for hate speech detection are able to outperform state-of-the-art char/word n-gram methods by nearly 18 F1 points. However, despite deep learning being at the forefront of hate speech classification, there still remains a lack of accurate deep learning models that can both detect instances of hate speech on Twitter and categorize them [4][5]. One of the most successful binary hate speech classification models reached an F1-Score of 84.83% and an AUC (Area Under the Receiver Operating Characteristic Curve) score of 90.39% [6][7]. The most successful multi-class toxic sentiment classification attempt reached an AUC score of 82% [8]. Additionally, only 51% of tweets violating Twitter guidelines are flagged by AI, while the other 49% have to be manually reported by other users. The methods behind these models, such as

David C. Wyld et al. (Eds): MLSC, ITCSS, ACSTY, SOFE, NATP, BDAB - 2023
pp. 93-103, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130207

CNN-LSTMs and the use of F1 and AUC scores as metrics served as inspiration for this project [9]. Furthermore, very little research has been done on the relationship between hate crimes, hate speech, and geographic locations of the incidents, which served as motivation for the choropleth map component of the project.

TweetWatch is a platform that automatically reports tweets marked as hate speech by passing real-time Twitter data through two novel deep learning models: a binary convolutional neural network (CNN) to detect hate speech and a multi-class CNN to classify hate speech into five categories: sexual orientation, special needs, gender, race, and other. Moreover, the solution includes an accessible, interactive choropleth map [10] of the United States created from the collected data. Previously, little effort has been made to find a correlation between geographical location and hate speech frequency, which TweetWatch solves using its innovative choropleth map. Furthermore, the deep learning models created for TweetWatch are significantly (over a 5% improvement) more accurate in terms of AUC and F1 scores.

To prove results, AUC and F1 scores were used to evaluate the accuracy of both models and select the best combination of batch size and epochs. First, we evaluated the reliability of the binary CNN using AUC and F1 Scores – were evaluated for 9 combinations of different batch sizes and epochs. Secondly, we similarity evaluated the reliability of the multi-class network using AUC scores, also for 9 combinations of different batch sizes and epochs.

The rest of the paper is organized as follows: Section 2 illustrates the details of the challenges faced during the span of the experiment; Section 3 focuses on the details of the methodology and the various components of the solution; Section 4 presents an analysis of the accuracy and viability of the solution, following by an evaluation of related works in Section 5. Finally, Section 6 provides concluding remarks, as well as points out possible future developments of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Lack of Annotated Data

As with many other supervised machine learning algorithms, one of the main challenges was finding sufficient annotated training data. Because of restrictions placed on the Twitter API, there is a lack of a consolidated, complete dataset of hate speech instances on Twitter. This problem is further exacerbated by the lack of annotated categorical hate speech datasets. To circumvent this problem, five different annotated datasets were combined to create a comprehensive dataset, and data points were manipulated to fit into each of the five categories, such as gender-based and sexuality-based discrimination. Data augmentation was also used to expand the set of training and testing data, facilitated by the Python nlpaug library.

2.2. Eliminating Biases

Another challenge with hate speech detection is dealing with societal nuances on Twitter. For example, marginalized communities often use demeaning jokes with each other and reclaim slurs for empowerment. Therefore, a common problem while dealing with hate speech detection is differentiating between non-harmful tweets and harmful tweets that often contain similar keywords. To solve this issue, extensive effort was used to make sure that training data included counterexamples of data that include hate speech keywords, such as slurs. This ensures that

context becomes important for the binary CNN as it learns to differentiate between hate speech and non-hate speech based on contextual phrases rather than specific words.

2.3. Constructing a Compact and Accessible Visualization Platform

Another one of the main challenges is constructing a compact visualization platform that is able to summarize and analyze the collected data in a compact, readable, and accessible format. Especially because one of the goals is to find a correlation between geographical location and hate speech frequency, a considerable challenge was to present this data in a graphical way. To solve this challenge, a choropleth map was created using Dash by Plotly to utilize color intensity and a continuous logarithmic scale to signify varying levels of hate speech frequency in different states. A pie chart that reconfigures itself based on user interaction was also created to facilitate a compact visualization of hate speech categorical breakdowns in each state.

3. SOLUTION

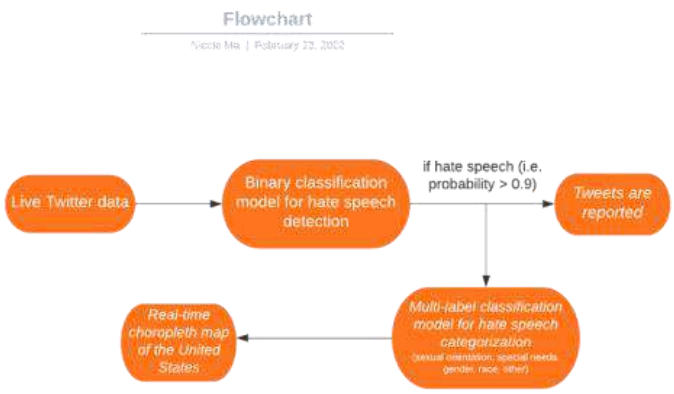


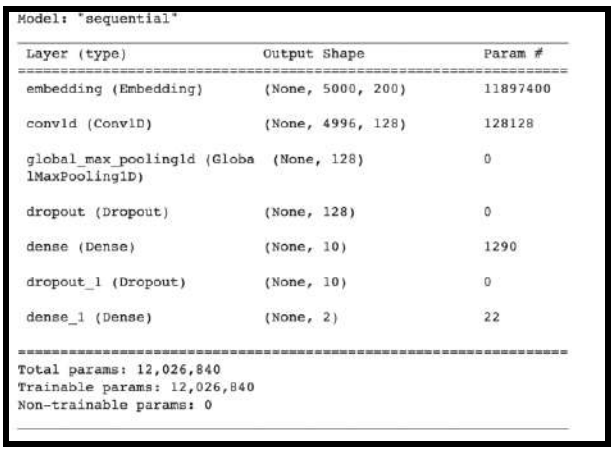
Figure 1. Overview of the solution

TweetWatch is a multi-step hate speech detection and categorization algorithm that automatically reports tweets marked as hate speech by passing real-time Twitter data through two novel deep learning models: a binary convolutional neural network (CNN) to detect hate speech and a multi-class CNN to classify hate speech into five categories: sexual orientation, special needs, gender, race, and other [11]. Natural language processing, such as tokenization and vectorization processes, is utilized to optimize the accuracy and efficiency of the neural networks. Through the use of dual neural networks, TweetWatch is able to go beyond simple identification of hate speech and also provide instant analysis of the frequencies of different categories of hate speech. Live Twitter data is scraped using the Twitter API, and tweets flagged as hate speech by the binary network are passed to Google Firebase [12]. Then, the multi-class network pulls data points from Google Firebase and categorizes the data, allowing TweetWatch to integrate the collected data into a publicly available, interactive choropleth map of the United States.

```
import json
class IDPrinter(tweepy.StreamListener):
  def __init__(self, time_limit=1):
    super(IDPrinter, self).__init__()
    self.start_time = time.time()
    self.limit = time_limit
    # super(IDPrinter, self).__init__()

  def on_status(self, tweet):
    if (time.time() - self.start_time) < self.limit:
      # statuses.append(eval(tweet)['text'])
      tweet_text = tweet.text.lower()
      tweet_text = tweet_text.replace("\n", "")
      tweet_text = re.sub('@[^\s]+','', str(tweet_text))
      tweet_text = re.sub('http[^\s]+' , '' , str(tweet_text))
      tweet_text = ''.join(filter(lambda x: x in printable, tweet_text))

      testing1 = model.predict([tweet_text])
      if testing1[0] >= 0.9 and len(tweet_text) >= 50:
        statuses.append(str(tweet_text))
        loc = tweet.user.location
        if loc != None and "," in loc:
          loc = str(loc)
          index1 = loc.index(",")
          twt_location1 = loc[(index1 + 2) :].lower()
          twt_location2 = loc[: index1].lower()
          for i in range(0, 98, 2):
            if twt_location1 == states_all[i] or twt_location1 == states_all[i+1]:
              location = states_all[i+1].upper()
              locations.append(location)
              break
              # locations_full.append(loc_full)
            elif twt_location2 == states_all[i] or twt_location2 == states_all[i+1]:
              location = states_all[i+1].upper()
              locations.append(location)
              break
            # if there is comma but not the rest
            else:
              location = "Invalid Location"
          if location == "Invalid Location":
            locations.append(location)
        else:
          locations.append("Invalid Location")
```

Figure 2. Web Scraping Live Twitter Data

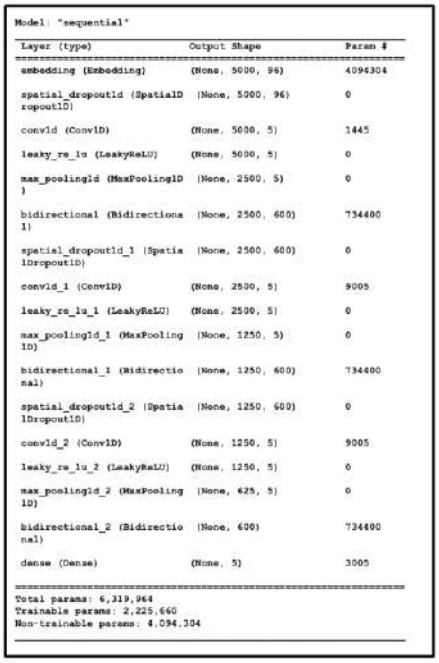
TweetWatch utilizes the TweePy library and the Twitter API to scrape real-time tweets, as well as information about the tweets, such as the location of the users. The scraper also standardizes the data, such as by removing links and reconfiguring emojis, to ensure that the format of the real-time data reflects that of the training data used for the convolutional neural networks. The collected data is passed to Google Firebase, where the data points are then sorted by properties such as location. New data from the web scraping algorithm is passed to Google Firebase every two minutes.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 5000, 200)         11897400

conv1d (Conv1D)              (None, 4996, 128)         128128

global_max_pooling1d (Globa  (None, 128)               0
lMaxPooling1D)

dropout (Dropout)            (None, 128)               0

dense (Dense)                (None, 10)                1290

dropout_1 (Dropout)          (None, 10)                0

dense_1 (Dense)              (None, 2)                 22

=================================================================
Total params: 12,026,840
Trainable params: 12,026,840
Non-trainable params: 0
_________________________________________________________________
```

Figure 3. Binary Classification Model

To create the binary convolutional neural network, a total of 40000 tweets were used to train the model. The data was de-biased by making sure there are counterexamples of data that contain hate speech keywords (e.g. slurs) and standardized by converting to lowercase and removing links, usernames, and non-ASCII characters using regular expression operations. Then standard NLP data pre-processing was utilized by fitting a Keras Tokenizer on collected tweets to split strings into tokens and using spaCy to create text embeddings. The final model compiles the model with the Adam optimizer and binary cross-entropy loss function and uses layers such as Conv1D, pooling, dropout, and dense.



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 5000, 96)          4094304

spatial_dropout1d (SpatialD  (None, 5000, 96)          0
ropout1D)

conv1d (Conv1D)              (None, 5000, 5)           1445

leaky_re_lu (LeakyReLU)      (None, 5000, 5)           0

max_pooling1d (MaxPooling1D  (None, 2500, 5)           0
)

bidirectional (Bidirectiona  (None, 2500, 600)         734400
l)

spatial_dropout1d_1 (Spatia  (None, 2500, 600)         0
l1DropoutID)

conv1d_1 (Conv1D)            (None, 2500, 5)           9005

leaky_re_lu_1 (LeakyReLU)    (None, 2500, 5)           0

max_pooling1d_1 (MaxPooling  (None, 1250, 5)           0
1D)

bidirectional_1 (Bidirectio  (None, 1250, 600)         734400
nal)

spatial_dropout1d_2 (Spatia  (None, 1250, 600)         0
l1DropoutID)

conv1d_2 (Conv1D)            (None, 1250, 5)           9005

leaky_re_lu_2 (LeakyReLU)    (None, 1250, 5)           0

max_pooling1d_2 (MaxPooling  (None, 625, 5)            0
1D)

bidirectional_2 (Bidirectio  (None, 600)               734400
nal)

dense (Dense)                (None, 5)                 3005

=================================================================
Total params: 6,319,964
Trainable params: 2,225,660
Non-trainable params: 4,094,304
_________________________________________________________________
```

Figure 4. Multi-class Classification Model

To train the multi-class convolutional neural network, more annotated datasets of Twitter hate speech were collected and de-biased. The datasets were manipulated datasets to fit into one or more pre-determined labels (0: sexual orientation, 1: special needs, 2: gender, 3: race, 4: other) and were concatenated horizontally into one Pandas Dataframe. The nlpaug library's synonym augmentation function was used to individually augment each dataframe to reach 12,000 tweets for each label (60,000 total), and a convolutional neural network was constructed using the leaky ReLu activation function and convolutional layers such as pooling and spatial dropout. The model was compiled with the Adam optimizer and categorical cross-entropy loss function.

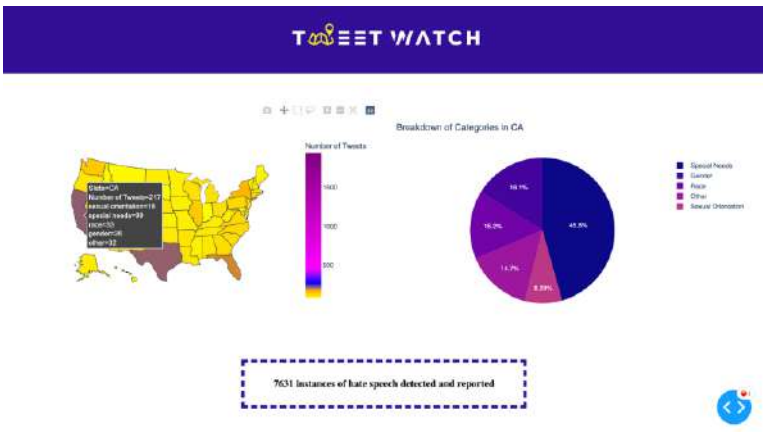


Figure 5. Choropleth Map

Using Dash by Plotly, data is collected from Google Firebase every two minutes and reconfigured into an interactive choropleth map of the United States. The choropleth map uses a logarithmic scale to measure the frequency of online hate speech in each state. By hovering over

a state on the choropleth map, users are able to view the corresponding breakdown of categories of hate speech in the state.

4. EXPERIMENT

4.1. Experiment 1

To evaluate the reliability of the binary convolutional neural network, two accuracy metrics – AUC and F1 Scores – were evaluated for 9 combinations of different batch sizes and epochs. A grid search was utilized to optimize the efficiency of the evaluation, and the obtained metrics of each epoch were recorded to construct training and validation curves.

Batch Size	Epochs	AUC (%)	F1 Score (%)
128	15	96.63	92.34
128	22	96.7	93.33
128	50	97.43	93.39
256	15	97.13	93.42
256	22	97.24	94.01
256	50	97.35	94.24
512	15	98.42	94.43
512	22	98.95	97.88
512	50	98.91	97.87

Table 1. AUC and F1 Scores for Binary Classification Model Trials

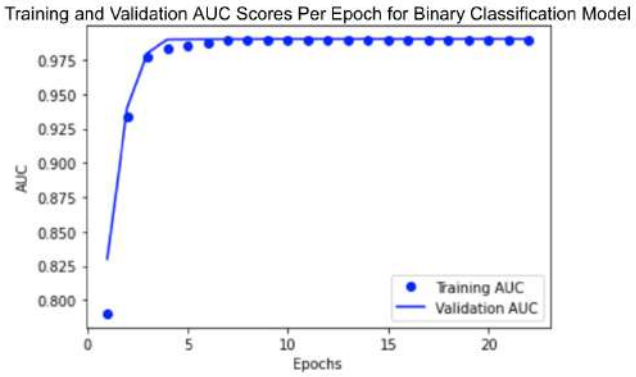


Figure 6. Graph of training AUC vs Validation AUC

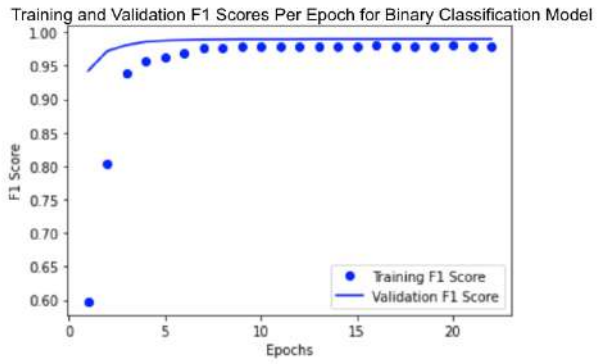


Figure 7. Graph of Training F1 Score vs Validation F1 Score

After grid searching, the most accurate binary classification model had an AUC score of 98.95%, an F1 score of 97.88%, and consisted of a batch size of 512 and 22 epochs. The reasoning behind the higher accuracy for 22 epochs, when compared to 50 epochs, is likely because the model overfitted between 22 and 50 epochs. The training and validation AUC and F1 curves for the best-performing binary model show a dramatic increase per epoch until it converges.

4.2. Experiment 2

Similar to the first experiment, 9 combinations of batch sizes and epochs were used to test the accuracy of the multi-class convolutional neural network with different parameters. Once again, a grid search of these combinations was used to analyze the accuracy of the model with respect to its AUC score, and the training and validation AUCs were recorded at each epoch to track the improvement of the model through the course of its training.

Batch Size	Epochs	AUC (%)
128	15	81.95
128	22	88.71
128	50	89.46
256	15	82.83
256	22	85.76
256	50	86.66
512	15	82.88
512	22	84.87
512	50	86.54

Table 2. AUC Scores for Multi-Class Model Trials

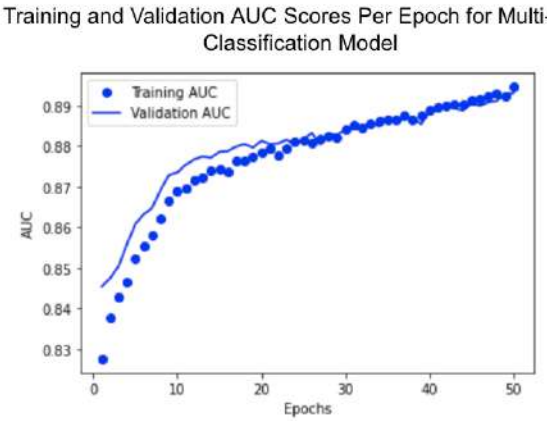


Figure 8. Graph of Training AUC and Validation AUC

After grid searching, the most accurate multi-class model had an AUC score of 89.46% and consisted of a batch size of 128 and 50 epochs. The training and validation AUC curves for the best-performing multi-class model show a steady increase per epoch.

Figure 9. Graph of AUC and F1 Score base, model, and target

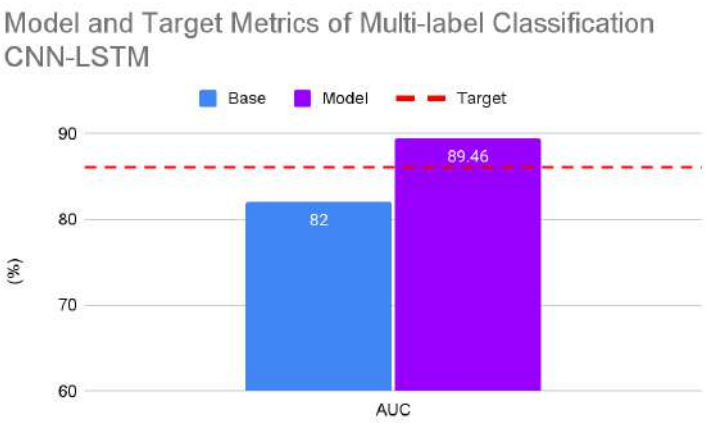


Figure 10. Graph of AUC base, model, and target

Metric	Binary Classification CNN	Multi-Class Classification CNN
AUC Score (%) / Increase from Baseline (%)	98.95 / 9.47	89.46 / 9.10
F1 Score (%) / Increase from Baseline (%)	97.88 / 15.38	N/A

Table 3. Summary of Best-Performing Model Variations

The final metrics for the binary classification model were 98.95% in terms of AUC score and 97.88% in terms of F1 score. These metrics surpassed the performance of models from multiple papers, including a 9.47% increase from the AUC of 84.83% and a 15.38% increase from the F1 score of 90.39% set as the baseline [6]. The final metric for the multi-classification model was 89.46% in terms of its AUC score, showing a 9.10% increase from the performance of the baseline model, which had an AUC Score of 82% [8]. All metrics surpassed the 5% target increase from the baseline models (See Section 1).

5. RELATED WORK

Carta et al. (2019) used a dataset of comments from Wikipedia's talk page to classify toxic comments [8]. To facilitate this, they used a supervised multi-class multi-label approach involving the Apache Spark big data framework and word embeddings to create a bag-of-word model. The AUC scores obtained from the model ranged from 0.71 to 0.75. Meanwhile, the multi-class CNN created for TweetWatch reached an AUC score of approximately 0.89, demonstrating an increase from the Carta et al.'s word embedding-based classification model.

Paul et al. (2018) created a set of neural networks to classify tweets as racist, sexist, or neither [6]. The study utilized GloVe embeddings after preprocessing the data by replacing items such as URLs with placeholder tokens. After testing a suite of machine learning models, such as BiLSTMs and CNNs, they found that the CNNs with the greatest reliability had an F1 score of 84.83% and an AUC/AUROC score of 90.39%. The binary CNN created for Tweetwatch was able to reach an F1 score of 97.88% and an AUC score of 98.95%, surpassing the accuracy metrics from the study.

Pereira-Kohatsu et al. (2019) created HaterNet to detect and monitor Spanish Twitter hate speech [13]. HaterNet utilizes the embeddings of words but also emojis and token expressions. Moreover, the analysis phase of HaterNet displays figures such as keyword frequency in tweets classified as hate speech. The best machine learning model from this study achieved an AUC score of 0.828. TweetWatch's binary AUC score of 0.9895 surpasses this value, and is more applicable to English-speaking countries than HaterNet.

6. CONCLUSIONS

Despite an increasing amount of hate speech on Twitter, there remains a lack of comprehensive deep-learning models that can both detect and categorize online hate speech. In this project, TweetWatch was created to serve as a comprehensive detection and reporting platform. TweetWatches utilizes two CNNs: a binary classification CNN to detect hate speech from real-time Twitter data and a multi-class CNN to categorize hate speech into five categories: gender, race, sexual orientation, special needs, and others. We used Dash by Plotly to create a real-time choropleth map of the United States with respect to frequency of Twitter hate speech in each American state. By using the AUC and F1 scores as metrics, we show that the novel deep learning networks are more reliable than previous models in multiple papers. The binary classification model had an AUC score of 98.95% and an F1 score of 97.88%. The multi-class classification model returned an AUC score of 89.46%. created.

Currently, the number of tweets able to be collected through web scraping is limited due to restrictions placed by the Twitter API. Moreover, many users do not have publicly available and accurately labeled locations, making it difficult to obtain a full understanding of region-based hate speech frequency. In contingence with this locational issue, TweetWatch is currently only able to create a choropleth map of the United States instead of the entire world to provide a more comprehensive understanding of global hate speech. Furthermore, the choropleth map shows hate speech frequency relative to each American state, but counties and other regions within each state might have varied frequency.

Future developments of TweetWatch will aim to mitigate these issues. The expansion of TweetWatch into a collaborative, cloud-based website running on users' devices would allow more tweets to be collected through web-scraping. Moreover, by utilizing a translation API and increasing the number of tweets collected through web-scraping, the choropleth map can be expanded to span the entire globe instead of just the United States. Furthermore, by improving upon the algorithm used to extract a user's publicly-available location could be improved on by adding variations of counties within each state.

REFERENCES

[1] Weller, Katrin. "Trying to understand social media users and usage: The forgotten features of social media platforms." Online Information Review (2016).
[2] Li, Qing. "Gender and CMC: A review on conflict and harassment." Australasian Journal of Educational Technology 21.3 (2005).
[3] Paz, María Antonia, Julio Montero-Díaz, and Alicia Moreno-Delgado. "Hate speech: A systematized review." Sage Open 10.4 (2020): 2158244020973022.
[4] Mosavi, Amir, Sina Ardabili, and Annamaria R. Varkonyi-Koczy. "List of deep learning models." International Conference on Global Research and Education. Springer, Cham, 2019.
[5] Bisong, Ekaba. Building machine learning and deep learning models on Google cloud platform. Berkeley, CA: Apress, 2019.
[6] Paul, Suvadip, and Jayadev Bhaskaran. "ERASeD: Exposing Racism and Sexism using Deep Learning." (2018).
[7] Lipton, Zachary Chase, Charles Elkan, and Balakrishnan Narayanaswamy. "Thresholding classifiers to maximize F1 score." arXiv preprint arXiv:1402.1892 (2014).
[8] Carta, Salvatore, et al. "A Supervised Multi-class Multi-label Word Embeddings Approach for Toxic Comment Classification." KDIR. 2019.
[9] Wang, Jin, et al. "Dimensional sentiment analysis using a regional CNN-LSTM model." Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers). 2016.

[10] Andrienko, Gennady, Natalia Andrienko, and Alexandr Savinov. "Choropleth maps: classification revisited." Proceedings ica. 2001.
[11] Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network." 2017 international conference on engineering and technology (ICET). Ieee, 2017.
[12] Chatterjee, Nilanjan, et al. "Real-time communication application based on android using Google firebase." Int. J. Adv. Res. Comput. Sci. Manag. Stud 6.4 (2018).
[13] Pereira-Kohatsu, Juan Carlos, et al. "Detecting and monitoring hate speech in Twitter." Sensors 19.21 (2019): 4654.