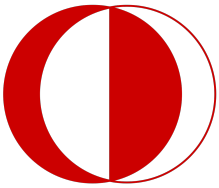

Middle East Technical University
Department of Computer Engineering

Detecting Turkish Troll Tweets

CENG790 Big Data Analytics
2020-2021 Fall
Term Project Report

Prepared by
Berker Acır & Erman Yafay
Student IDs: 2098697 & 2205839
Computer Engineering
31 January 2021

Abstract

We train a classifier to detect Turkish *troll* tweets by only using the tweet itself, i.e. we do not incorporate any account specific information. Our best model configuration achieves 89.7% accuracy on a test dataset of 3147 samples and can be used to classify any sentence-like input. Therefore, it is not limited to only Twitter data, but can be used in similar social media content such as Facebook or Reddit comments.

1 Introduction

The spread of misinformation through social media is an increasing concern for the internet users. Bot nets or actual users are used to spread the misinformation on demand. One of the adversarial affects of such a spread is that it allows large population of crowds to follow a certain action such as political decisions. A prime example of such a case is the Russian interference on U.S. elections[1]. If we were to let the misinformation roam freely, the people who can spread such propaganda at a large scale could create a danger for the reputation of individuals and institutions. Moreover, in a more severe case, the social media would be cluttered with propaganda and the trustworthiness of it as a news source would be damaged.

Such kind of manipulation has been widely observed in Twitter. As a result of this, Twitter took an action [1] and released a network of state-backed *troll* users from Iran, Turkey and Russia together with their tweets and other related data such as media. The purpose of these users is to act collaboratively on matters where the state wants to promote certain ideas for its advantages. Therefore, we aim to use this data for not only to detect state-backed users, but also to obtain knowledge about the underlying semantics of generating *troll* content such that this knowledge could be used for any social media.

In Section 2 we describe our dataset, show some example of tweets and also describe how we processed it for our purposes. Then in Section 3, we describe our training strategy and Section 4 presents our results followed by a discussion. Finally, we conclude this report with Section 5.

2 Dataset

In order to train our classifier, we obtained the *troll* and *non-troll* data from the following data-sets respectively:

- Official Twitter State-backed tweets [1]
- Archived Twitter stream by Archive Team [2]

The former dataset has been officially released by Twitter where state-backed users have been identified and their tweets have been published. The latter one contains all of the tweets that can be extracted from the Twitter API in which we've used it to extract our *non-troll* tweets. Since the two datasets were different in terms of field names and count, data types and structures, we had to process them separately in order to get similar datasets for both *troll* and *non-troll* tweets. Also, the datasets had to be filtered such that there were only Turkish, not retweets from same period of time. Therefore, the followings pre-processing steps were applied on the datasets in order to obtain *troll* and *non-troll* tweets for:

1. Filtering tweets:
    - tweeted from January 2020
    - in Turkish language
    - not retweeted
    - tweeted by verified user (only for non-troll tweets)
2. Reduce the number of fields

[1] `https://en.wikipedia.org/wiki/Russian_interference_in_the_2016_United_States_elections#Social_media_and_Internet_trolls`

- Tweet related fields: text, time, hashtags, URLs, user mentions
- User related fields: profile description, follower count, following count, account creation date

Due to the large size of the archived Twitter stream, for example a single month requires more than 60GB of storage, we decided to choose a specific time period, namely January 2020, to prepare the data for training. Our choice in date is not random, as we observed that the state-backed troll users were active the most in January 2020. As a result of this pre-processing we obtained the *troll* and *non-troll* datasets.

2.1 Troll Tweets Dataset

Twitter disclosed state-backed accounts as one of the responsibilities of Twitter is to protect the integrity of the public conversation [3]. In June 2020, Twitter released the dataset that contained disclosure information of state-backed entities in Turkey. In this project, we use the anonymized version of the dataset. It is also possible to request the identified version of the dataset but the anonymized version is sufficient for our purposes.

In total Twitter disclosed 7340 accounts with the in following categories:

- Account Information (533 KB)
- Tweet Information (5 GB)
- Media (821 GB, 391 archives)

For our purposes, we only used Tweet Information category as it contained troll accounts' information and our methodology does not require additional information such as media. Tweet information dataset has 30 fields and we use the following fields:

- **user_profile_description** - the user's profile description (*)
- **follower_count** - the number of accounts following the user (*)
- **following_count** - the number of accounts followed by the user (*)
- **account_creation_date** - date of user account creation
- **tweet_language** - the language of the tweet
- **tweet_text** - the text of the tweet (mentions of anonymized accounts have been replaced with anonymized userid)
- **tweet_time** - the time when the tweet was published (UTC)
- **is_retweet** - True/False, is this tweet a retweet
- **hashtags** - a list of hashtags used in this tweet
- **urls** - a list of urls used in this tweet
- **user_mentions** - a list of userids who are mentioned in this tweet (includes anonymized userids)

  (*) - at the time of suspension

State-backed tweets dataset contained 36.948.537 state-backed tweets dating back to 2009. After filtering, we obtained 219.131 *troll* tweets, which are Turkish, not retweets and tweeted in January 2020.

2.2 Archived Twitter Stream

Preliminary to our pre-processing the stream contained 153.093.990 tweets in the January 2020. After applying the same filter with the state-backed tweets we obtained 1.100.096

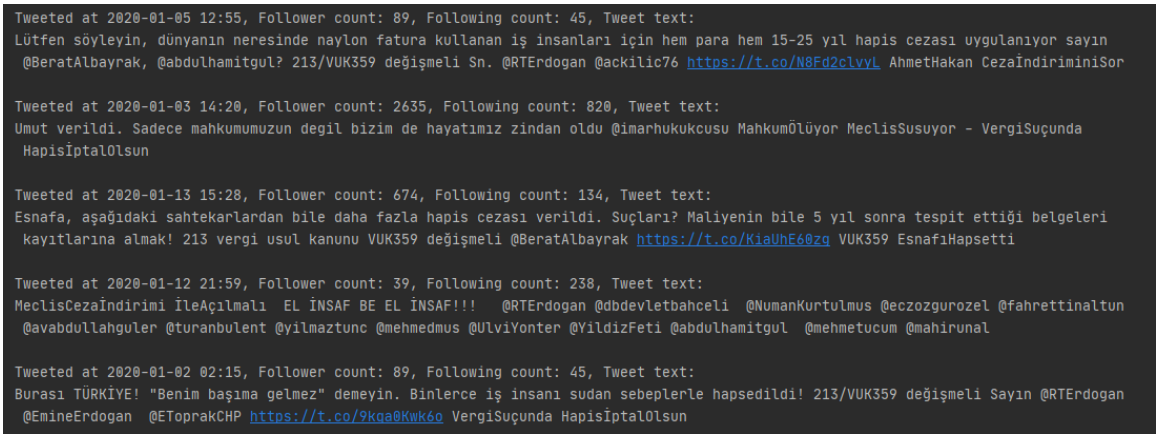

Figure 1 . Example tweets from troll tweets dataset.

tweets, which were Turkish and not retweets. For archive stream we used an additional constraint; tweets need to come from verified users. In this way, we obtained 6294 *non-troll* tweets which required us to process more than 60GB of data. We also used the same time period as the *troll* tweets i.e. we use the archive stream in January, 2020.

Schema of archive stream dataset had more than 1300 fields that is because tweets can be nested inside a tweet. We used the following attributes of Tweet object [4]:

- **created_at** (*String*): UTC time when this Tweet was created.
- **text** (*String*): The actual UTF-8 text of the status update.
- **truncated** (*Boolean*): Indicates whether the value of the text parameter was truncated, for example, as a result of a retweet exceeding the original Tweet text length limit of 140 characters. Truncated text will end in ellipsis, like this ... Since Twitter now rejects long Tweets vs truncating them, the large majority of Tweets will have this set to false. Note that while native retweets may have their top-level text property shortened, the original text will be available under the retweeted_status object and the truncated parameter will be set to the value of the original status (in most cases, false).
- **user** (*User*): The user who posted this Tweet.
- **retweeted_status** (*Tweet*): Users can amplify the broadcast of Tweets authored by other users by retweeting . Retweets can be distinguished from typical Tweets by the existence of a retweeted_status attribute. This attribute contains a representation of the original Tweet that was retweeted. Note that retweets of retweets do not show representations of the intermediary retweet, but only the original Tweet. (Users can also unretweet a retweet they created by deleting their retweet.)
- **entities** (*Entities*): Entities which have been parsed out of the text of the Tweet.
- **lang** (*String*): Nullable. When present, indicates a BCP 47 language identifier corresponding to the machine-detected language of the Tweet text, or 'und' if no language could be detected.

We used the following attributes of User object [5]:

- **description** (*String*): Nullable. The user-defined UTF-8 string describing their account.
- **verified** (*Boolean*): When true, indicates that the user has a verified account.

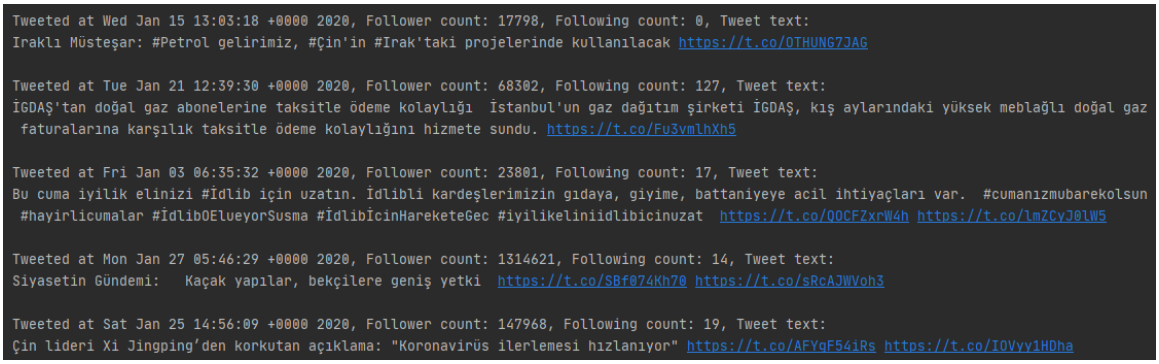

Figure 2 . Example tweets from non-troll tweets dataset.

- **followers_count** (*Int*): The number of followers this account currently has. Under certain conditions of duress, this field will temporarily indicate "0".
- **friends_count** (*Int*): The number of users this account is following (AKA their "followings"). Under certain conditions of duress, this field will temporarily indicate "0".
- **created_at** (*String*): The UTC datetime that the user account was created on Twitter.

We used the following attributes of Entities object [6]:

- **hashtags** (*Array of Hashtag Objects*): Represents hashtags which have been parsed out of the Tweet text.
- **urls** (*Array of URL Objects*): Represents URLs included in the text of a Tweet.
- **user_mentions** (*Array of User Mention Objects*): Represents other Twitter users mentioned in the text of the Tweet.

3 Methodology

Since we have a large discrepancy between the number of *troll* and *non-troll* samples, we limit the sample size of both classes to the smaller one i.e. to 6294 samples for each class.

Our approach uses two different pre-trained BERT models for Turkish [7] to capture the semantic content of the tweets. These models do not operate at the word level but rather each word in a sentence (a tweet in our case) is also dependent on other words such that the output of these models are sentence sensitive [8]. These models are highly complex and trained on very large corpora. Therefore, creating such a model from scratch is out of our concerns at the scale of this project[2].

We experiment with two different pre-trained models; first one is a distilled, case sensitive (cased) BERT model and the second one is a case insensitive (uncased) one. For the sake of our experiments we would like to change one experiment configuration parameter at a time e.g. changing from cased to uncased. However, we needed to use an undistilled version of the uncased model since it is not available to us at the time of this writing.

3.1 Just the Tweet

As we also mentioned in the earlier sections, we only use the text field of the tweets. This approach has the advantage of context insensitivity i.e. we do not need to acquire account

[2] `https://github.com/CENG790-GROUP5/tr-troll-detect`

Table 1. Hyper-parameters for logistic regression models

Hyper-parameter	Description	Values
`regParam`	Weight of the regularization error	$0.1, 0.01, 0.001$
`maxIter`	Maximum number of iterations for the solver	$50000, 100000$
`threshold`	Probability threshold to classify as troll	$0.5, 0.7, 0.9$

specific information. With the trained model, we can simply classify tweets from anywhere using only the tweet itself or any other kind of sentence-like string. In this manner, the trained model can be used to detect trolls on other social media as well. Additionally, tweet text is obviously the main feature in our data. It is the largest portion of the information. A classifier should not decide solely on account specific information. On the other hand, further investigation on the affect of additional features is a matter that needs to be investigated further.

3.2 Tweet to Input

To use the pre-trained models we first tokenize the input text using the WordPiece tokenizer [9] that is trained alongside with the BERT models [7]. WordPiece starts with an initial vocabulary of all characters in the training data, then iteratively learns to combine tokens such that the likelihood of generating the training data using the vocabulary is maximized. Later the tokenized tweets are forwarded through the BERT model to obtain the final 768 dimensional input vector.

3.3 Training

We follow a simple approach; we use logistic regression on the output of the BERT model to train a binary classifier. We couple the BERT models with the logistic regression classifier and obtain two different models, namely `DistBERT-C-Log` is the distilled, cased BERT version and the `BERT-U-Log` is the undistilled, uncased version.

For each model we split three quarters of our data for training and the rest for testing in a random manner. This constitutes 9441 training and 3147 test samples. We then further reserve a 20% of our training data for validation of our hyper-parameters that we present in Table 1.

4 Results

In Table 2 we present the evaluation of our models. Tuned parameters are given in the order of `regParam`, `maxIter` and `threshold`. Our results show that even the tweet text itself is sufficient to train a useful classifier as we achieved 84.5% and 89.7% test accuracy on our test data. Since training and test accuracy are similar, both models generalize well. We also achieve similar results for the F-Score which shows that our models have a good balance between false positives and negatives. Since we also have equal number of samples from each class, accuracy itself is useful enough for our evaluation purposes.

Although `BERT-U-Log` is an undistilled model, it outperforms `DistBERT-C-Log`. Therefore, case insensitivity increases classifier accuracy. An interesting take away from our tuned parameters is that the `threshold` parameter. We obtain best validation error when we use high (0.9) probability threshold for `DistBERT-C-Log` and low (0.5) for `BERT-U-Log`. This

Table 2. Train-test accuracies, test F-Scores together with tuned parameters for each model

Model	Train Acc.	Test Acc.	Test F-Score	Tuned Parameters
`DistBERT-C-Log`	0.839	0.845	0.841	$0.001, 100000, 0.9$
`BERT-U-Log`	0.920	0.897	0.897	$0.01, 100000, 0.5$

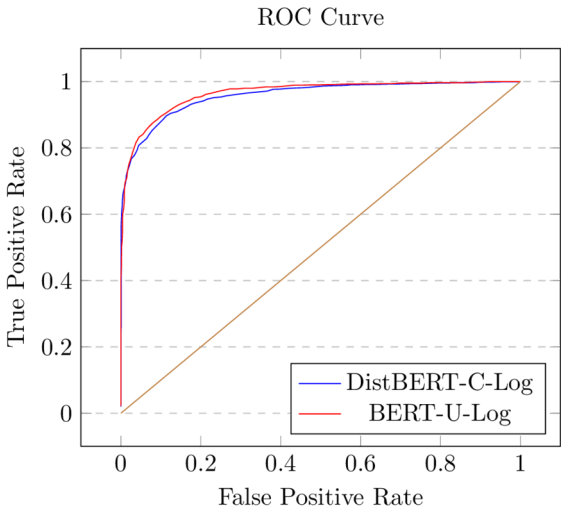

Figure 3 . ROC Curve for our logistic regression models

shows that the cased model needs to be more conservative while classifying tweets as *troll*. ROC curves in Figure 3 also show that `BERT-U-Log` performed better for all threshold values.

5 Conclusion

In this project, we trained and tuned a classifier for detecting Turkish troll tweets by just from the tweet's text and the results show that the best model achieves 89.7% accuracy on test dataset. Different BERT models are used for tuning the model along with logistic regression hyper-parameters.The model simply turns tweet texts into input vectors with Turkish BERT model and feeds the vector into logistic regression classifier.

Work done in this project can be expanded to other social platforms as only a text is used as an input for detecting whether it is troll or not. It can be further improved for Twitter case with the use of additional input parameters such as user's follower and following count, tweet time, hashtags, URLs, mentions.

References

[1] Twitter, ''Disclosing networks of state-linked information operations we've removed,'' Jun 2020. [Online]. Available: https://blog.twitter.com/en_us/topics/company/2020/information-operations-june-2020.html

[2] ''Archive team: The twitter stream grab,'' Dec 2012. [Online]. Available: https://archive.org/details/twitterstream

[3] ''Information operations,'' Oct 2018. [Online]. Available: https://transparency.twitter.com/en/reports/information-operations.html

[4] ''Tweet object.'' [Online]. Available: https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/tweet

[5] ''User object.'' [Online]. Available: https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/user

[6] ''Entities object.'' [Online]. Available: https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/entities

[7] S. Schweter, ''Berturk - bert models for turkish,'' Apr. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3770924

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''Bert: Pre-training of deep bidirectional transformers for language understanding,'' *arXiv preprint arXiv:1810.04805*, 2018.

[9] M. Schuster and K. Nakajima, ''Japanese and korean voice search,'' in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 5149--5152.