Supervised Classification of Twitter Accounts Based on Textual Content of Tweets

Fredrik Johansson
fredrik.johansson@foi.se

PAN @ CLEF 2019

September 10, 2019

Outline

- A security and intelligence perspective on bot and gender profiling
  - Motivation and examples
  - Our previous work (mostly metadata-based)
- Implemented two-step binary classification approach
  - Features and classifiers
  - Results

Information operations in social media

Social media used by e.g. state actors to carry out various types of *information operations*

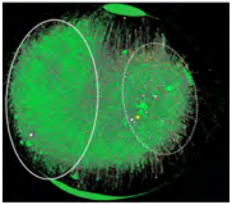
- Bots: "Drown" hashtags with unrelated content, information spread (trending topics), manipulate reputation statistics . . .
- Trolls: increase tension and polarization in societies (NATO, migration, Brexit, gun control, etc.)
- Hi-jacked accounts: make use of existing accounts' social network and reputation to reach out to large audience (e.g., hi-jacking of @AP)

Detection of Twitter bots

- Divert attention from protests by flooding hashtags: e.g., Syria, Mexico, Russia
- Amplification of messages: e.g., accounts depicting Hong Kong protesters as violent criminals
- Growing threat with improved neural models for text generation, such as GPT-2 and Grover
    - Increased automation of troll activities?

Tools for analyzing information operations on Twitter

- Visual analytics for identifying coordinated accounts

- NLP object patterns for detecting tweets of interest
  - E.g., "Lavrov and Putin propaganda machine are on overdrive today"
- Automatic classification of bots
  - E.g., inter-tweet content similarity, inter-tweet timing distributions, inter-tweet delay regularities, # hashtags, # mentions, # URLs

Gender profiling

In criminal investigations or intelligence work, profiling anonymous accounts can sometimes be of importance

- Example: Death threats sent to politicians to their home adresses (with related searches conducted from a certain IP address)
- Profiling gender or other characteristics can sometimes decrease number of likely senders
  - Use of function words, POS tags etc.
  - Does not seem to work very well for Twitter data!

FOI

High-level approach

- Two-step binary classification
  1. Bot or human?
  2. Male or female? (only if classified as human)
- Calculate aggregate statistics based on all tweets from account of interest
  - Signs of bots which are not visible on individual tweet level
  - E.g. inter-tweet similarity

Aggregate "metadata" statistics (bot classification)

Calculate *max*, *min*, *avg*, *std* for the following features:

```python
def calculateStats(df):
    print("Starting to calculate stats")
    df['str_len'] = df.content.str.len()
    df['nr_of_mentions'] = df.content.str.count("@")
    df['retweet'] = df.content.str.contains("RT ")
    df['link'] = df.content.str.contains("http")
    df['nr_upper'] = df.content.str.count(r'[A-Z]')
    df['nr_lower'] = df.content.str.count(r'[a-z]')
    df['content_shifted'] = df.groupby(['id'])['content'].shift(1)
    df['content_shifted'] = df["content_shifted"].replace(np.nan, '', regex=True)
    df['edit_distance'] = df.apply(get_text_dist, axis=1)
    return df
```

Damerau-Levenshtein used as edit distance metric on adjacent tweets.

Content features (bot classification)

Aim at simplicity/generalizability rather than optimizing dev-set performance

- Concatenate all tweets for current user
- Apply TfidfVectorizer in scikit-learn
  - analyzer = "word", lowercase = True
  - ngram_range = (1,2), max_features = 800
  - min_df = 4, binary = True (TF-part 0 or 1)
  - use_idf = True, smooth_idf = True
- LSTMs or Transformers with pre-trained word embeddings would be more powerful, avoided due to TIRA performance and need for scaling to large datasets in our tools

Bot classifer

Trained separate classifiers for TF-IDF and the "metadata" features, due to relative sparseness of TF-IDF vector

1. Logistic regression classifier on the TF-IDF features
   - Regularization: C=1.0
2. Add output class probabiilties from log. reg. as additional feature
3. Random Forest classifier on statistical features + log. reg. output
   - n_estimators=500
   - max_features="auto"
   - min_samples_leaf = 1

Grid search was used on training set to select classifiers with suitable parameter settings

Gender classifer

Ended up with extremely simple gender classifier

- Logistic regression classifier on based on most common TF-IDF features in training data
    - Regularization: C=1.0
    - TF-IDF
        - analyzer = "word", lowercase = True
        - ngram_range = (1,1), max_features = 300
        - min_df = 10, binary = False
        - use_idf = True, smooth_idf = True
- Experimented with the statistical features, POS tags etc. but did not increase performance

Results

Task	**Lang**	**Dev set**	**TIRA testset2**	*Rank**
Bots profiling	en	0.948	0.960	Top-1
Bots profiling	es	0.892	0.882	Top-15
Gender profiling	en	0.752	0.838	Top-5
Gender profiling	es	0.648	0.728	Top-20

* 55 participating teams in total

Consistently underperform on Spanish compared to English. Used default string tokenizer in scikit-learn, probably a terrible idea...

Questions?

Thanks for listening!

fredrik.johansson@foi.se