HinReddit

- Chengyu Chen
- Yu-chun Chen
- Yanyu Tao
- Shuibenyang Yuan

1. Hateful Post Classification

As countless social platforms are developed and become accessible nowadays, more and more people get used to posting opinions on various topics online. The existence of nagetive online behaviors such as hateful comments is also unavoidable. These platforms thus become prolific sources for hate detection, which motivates large numbers of scholars to apply various techniques in order to detect hateful users or hateful speeches.

In our project, we plan to investigate contents from Reddit, which is a popular social network that focuses on aggregating American social news, rating web content and website discussion, that carries rich potential information of contents and their authors. Our goal is to classify hateful posts from the normal ones. Being able to identify hateful posts not only enables platforms to improve user experiences, but also helps to maintain a positive online environment. **We would like to stress that the boundary of 'hate' is vague and there is no correct nor consolidated definition of 'hatefulness,' our classification of hateful posts depends only on a unified definition within our team, which we divide into the categories of `severe_toxic`,`threat`, `insult`, and `identity_hate`.** We all agree that other people's recognition of "hate" may be but not limited to these four categories, and our labeling method allows full freedom of other definition of "hatefulness."

We plan to use Bidirectional Encoder Representations from Transformers (BERT), a neural network architecture transforming natural language processing (NLP) techniques, in our data ingestion pipeline for data labeling. However, instead of using NLP in attempts to solve classification problems, we will be using graph embedding methods. Specifically, we will create a heterogeneous information network to capture the relationships among Reddit posts, which is then used as our features.

If our project is successful, we will have built an application, *hinReddit*, which helps identify hateful posts for Reddit. Similarly, others can apply our process on different social platforms. In addition, we will create a blog post including an EDA on the data we extracted and detailed description of the process we will complete to

ingest data. We will perform feature engineering, develop a neural network model, and finally a summary of the test result of our model.

2. Related Works

Hindroid

Detecting hateful posts on Reddit is similar to our domain problem of detecting Android malware both conceptually and technically. Despite using different platforms, these two case studies both aim at identifying the malicious units from the benign units, and the goals are to produce a healthier and more positive environment to users. As we did in our replication using graph embedding techniques, here in our study, we will also pay attention to the connections as well as the communities of our object and construct heterogeneous information network (HIN) upon those connections that enables further training and classifications.

Specifically, in our HIN graph, we will have Reddit post nodes equivalent to App nodes in the replication project and user-interaction nodes equivalent to API nodes in the replication. While Hindroid investigates more of the relationships among API calls, for instance, having three out of four matrices developing different interactions of APIs, and thus focuses less on relationships among Apps themselves, we plan to add to our HIN the relationship among Reddit post nodes themselves to further diversify our network graph.

Social Network Based Problems

Studies regarding the detection of hateful speech, content, and user in Online Social Networks have been manifold. In the report Characterizing and Detecting Hateful Users on Twitter, the authors present an approach to characterize and detect hate on Twitter at a user-level granularity. Their methodology consists of obtaining a generic sample of Twitter's retweet graph, finding potential hateful users who employed words in a lexicon of hate-related words and running a diffusion process to sample more hateful users who are closely related in the neighborhood to those potential ones. However, there are still limitations to their approach. Their characterization has behavioral considerations of users only on Twitter, which lacks generality to be applied to other Online Social Networks platforms. Also, with ethical concerns, instead of labeling hate on a user-level, we want to avoid tagging individuals and believe that detecting hate on a content-level will be more impartial.

3. The Data

Dataset

Our project includes two datasets:

1. Main dataset used for our project analysis This is a dataset we will obtain from Reddit through a couple APIs. We use the API called PushShift to obtain Reddit post information, including post text, title, and user ids who reply to either the post itself or any of the reply below the post and the comments that it provided. We use `PushShift` because it offers a specific API to obtain the flattened list of repliers' ids and takes considerably less time than doing the same with PRAW. After a brief EDA on the most popular 124 subreddits, we select 50 subreddits in which the proportion of valid text posts of the posts are the highest and then sample a number of newest posts in each of the 50 subreddits. By doing this, our data will represent a population of newer posts in subreddits whose posts have higher text-proportion. We want to eliminate image/meme posts and deleted posts so we can better apply NLP model for our supervised learning.

    - advantages:
        - This dataset is obtained from the actual social platform, and thus we obtain real-world perspective when training.
        - Reddit has a couple APIs for us to suit our different needs.
    - limitations:
        - There are no ground-truth labels we can use for the data we collect, and thus need the assistance of other well-defined and pre-trained models to first label our data.
        - We are not certain of the level of hatefulness from Reddit posts we obtain, and may lead to an unbalanced number of posts in benign and hateful categories.
        - Our dataset will include newest posts in each subreddit, and may not apply well for older posts.

2. Kaggle Toxic Comment Classification Dataset This is a dataset provided on https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data, including information of hundreds of thousands of wikipedia comments along with multiple negative labels. We will be mainly using this dataset to train a nlp pretrained BERT classifier model to label our reddit post data before we use it for HIN learning.

    - advantages:
        - This dataset is labeled, allowing us to perform supervised learning to train a nlp classifier model.
        - The dataset include several labels, including `severe_toxic`, `obscene`, `threat`, `insult`, `identity_hate`, thus giving us some space to define what constructs a hateful post.
    - limitations:
        - We are not certain if labels for wikipedia comments can be applied to posts from Reddit or other social platforms.

Data Ingestion Process

Data Origination and Legality

1. Our data entirely originates from Reddit. We will be using the Reddit APIs to obtain the data from the website. As stated in Reddit's API Terms of Use, in order to legally use the Reddit API, it is necessary for us to agree with all the applicable policies and guidelines listed in the Terms of Use. With a careful review of the document, we understand that we have satisfied all requirements and grant consent on all Terms. Moreover, since we have registered Reddit accounts agreeing with all terms and conditions, we believe our usage of the Reddit API is legal.

2. The Kaggle Toxic Comment Classification data originates from the comments of Wikipedia's talk page edits and is distributed through a closed Kaggle competition. According to the Competition Rules, for the specific "competition data", or the datasets available from the the Competition Page for the purpose of use in the Competition, users are allowed to access or use the data for academic research and education, or non-commercial purposes. Our usage of the data will not violate the rules.

Privacy Concerns

As Reddit is an online public social platform and all posts and replies are open to viewers, we will not get into issues regarding privacy. Nevertheless, we will encrypt all users' personal information if involved and eliminate

sensitive posts or replies in case of any information leakage.

Schema (updated)

The data schema is shown below

```
./data
├── interim
│   ├── graph
│   │   ├── graph.mat
│   │   └── processed
│   │       ├── graph.pt
│   │       ├── pre_filter.pt
│   │       └── pre_transform.pt
│   └── label
│       ├── comment
│       │   ├── buildapc.csv
│       │   ├── legaladvice.csv
│       │   └── ...
│       ├── label.csv
│       └── post
│           └── post_sentimental.csv
├── processed
│   └── node2vec
│       ├── data.pt
│       ├── embedding.pt
│       └── log.json
└── raw
    ├── comments
    │   ├── buildapc.csv
    │   ├── legaladvice.csv
    │   ├── log.json
    │   └── ...
    ├── posts
    │   ├── buildapc.csv
    │   ├── legaladvice.csv
    │   ├── log.json
    │   └── ...
    └── posts_detail
        ├── buildapc.json
        ├── legaladvice.json
        └── ...
```

Raw

First Layer: Posts

The csv file contains the information of each post in a dataframe where the unit of observation is the individual post.

- `id`: post_id
- `author`: username of the author who make the post
- `title`: title of the post
- `selftext`: the content of the post
- `num_comments`: number of comments
- `created_utc`: the epoch date for which the post is created
- `full_link`: the link to the reddit post
- `subreddit`: subreddit it belongs to
- `score`: number of upvote - number of downvote

Second Layer: Post detail

The file contains certain number of posts id and all of its comments id under a certain subrredit.

- `submission_id` : id of the post
- `comment_ids`: id of each comment


```
[{"submission_id":"fsoala","comment_ids":[]},
{"submission_id": "fsnmj4", "comment_ids": ["fm2fd48", "fm2hrmh",
"fm2k37i", "fm2k8p4", "fm2kuot", "fm2lces", "fm2lsao", "fm2lu4n",
"fm2m5at", "fm3trkl", "fm4c7i6"]}]
```


Third Layer: Comments

The csv file contains the information of each specific post in a dataframe where the unit of observation is the individual comment.

- `id`: comment id
- `author` : username of the author who make the comment
- `created_utc` : the epoch date for which the comment is made
- `is_submitter`: whether that person post the original post
- `subreddit`: the subreddit it belongs to
- `link_id`: the post id for which this comment is made for

Interim

Graph

- `graph.mat` contains the sparse matrix file of N, P, U, A matrices
- `processed` directory contains the pytorch datasets of `graph.mat`.

Label

- `label.csv` contains the label of posts after taking in to account comment labels.
- `comment` directory conatians sentimental analysis over comments by different subreddits.

- `post` directory contains sentimental analysis over posts.

Processed

- `node2vec` directory contains the pytorch embedding layer of node2vec algorithm.

Data Cleaning

Since we are directly using the Pushshift API, it has taken care most of the data cleaning parts. Since the output of the result is in .json format, thus the only transformation we have to make is to use pandas to output the result in .csv format.

Applicability

The above data ingestion pipeline can be used to obtain data as long as the data originates from Reddit. Our pipeline has limited applicability depending on data sources. Possible data sources include other online social platforms such as Twitter, Facebook, LinkedIn, and Instagram. Platforms have similar overall structure but differ in detailed construction and API calls, thus our pipeline may only be helpful for general data ingestion framework reference when applying to other online social platforms. Also, it is important to check the policies and guidelines of each platform before employing our pipeline to avoid the raise of legal issues or privacy concerns.

4. Labeling (updated)

Since the original data obtained from Reddit is not labeled, we will be using a RNN and bidirectional layers, through python library `keras`, as well as pre-trained word representation vectors from GloVe, to label the Reddit posts before we use it for our project main analysis.

By following a tutorial of using `keras` and the pretrained word vectors, we will train a multi-label NLP model with kaggle labeled dataset of wikipedia comments detailed in Datasets. We will save this model in directory `interim`. This multi-label model then can be used to calculate each Reddit post or comment a score between 0 to 1 for each of the label `toxic`, `severe_toxic`, `threat`, `insult`, `identity_hate`.

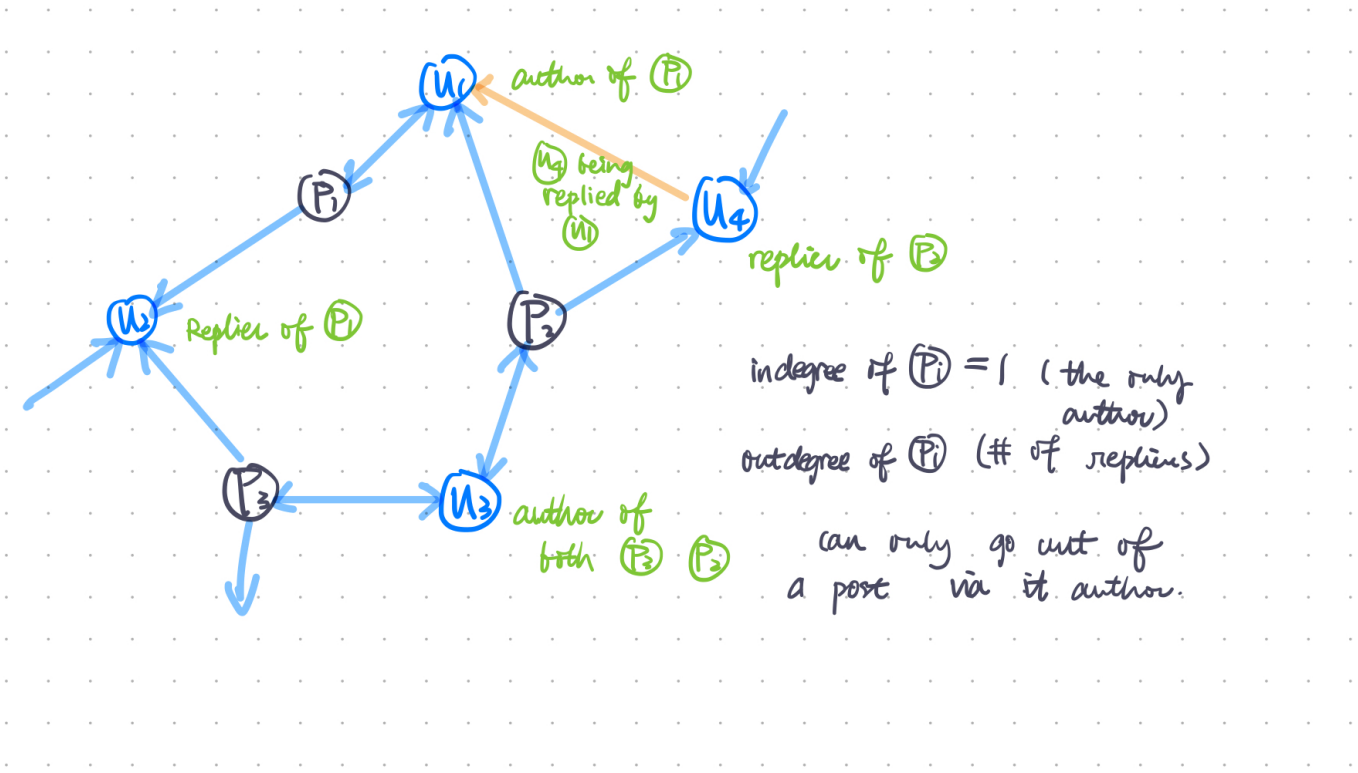
The labeling process for a single post is as follows: We first obtain scores for the post itself if it has textual content. Then, we also obtain scores for all of its comments to compute an average of all five labels. We then compute total scores for the post by adding scores of post content and its comments. We then compute the max of all five total scores, and if the max value is greater than the threshold, we classify the post as 'hateful'. In our project, we set the threshold to 0.5. If the post is removed it will be labeled as deleted and the NA post will also be labeled as NA.

In this way, we can also label those posts which are missing textual content by making use of its comment data. Moreover, with this labeling process, we are defining hateful posts so that they not only include those that demonstrate hatefulness in its content, but also those that stir up negative discussions in comments and replies.

5. Graph Extraction (new)

Graph Structure

the graph structure is shown as below:



The graph consists two kind of nodes:

- Post Node: the post group
- User Node: the user group
  - Author Node: author of the post
  - Commentor Node: commentors who answer the post or comments under the posts
  - Note: Author Nodes and Commentor Nodes can be overlapped

The graph rule is explained as following:

- Post Nodes can go to all User Nodes under them directly (both Author and Commentor).
- Author Nodes can only go to Post Nodes they associate with.
- Commentor Nodes can only go to User Nodes who reply them.

Adjacency Matrix

We will represent our Graph into following Adjacency matrix form:

```
- U matrix
  - if user i has been replied by user j, then i,j entry of U will be
added 1

- P matrix
  - if post i has author or commentor j, then i,j entry of P will be added
1

- A matrix
  - if author i writes post j, then i,j entry of A will be 1
```

```
- N matrix
  - N matrix will be the homogeneous representaion of hetergeneous graph
above
  - U | A
    P | 0
```

Graph Example

```
we have two reddit posts with id 1 and 2 shown below

post_id: 1
author_user_id: 1
  commentor_user_id: 2
    commentor_user_id: 3
post_id: 2
author_user_id: 2
  commentor_user_id:4
  commentor_user_id:2

The Matrices will be:

U:  0|0|0|0
    0|0|1|0
    0|0|0|0
    0|1|0|0

P:  1|1|1|0
    0|1|0|1

A:  1|0
    0|1
    0|0
    0|0

N:  0|0|0|0|1|0
    0|0|1|0|0|1
    0|0|0|0|0|0
    0|1|0|0|0|0
    1|1|1|0|0|0
    0|1|0|1|0|0
```

6. EDA (updated)

Tabular Data EDA

As you may know, Reddit has already banned lots of subreddit that contained explicit or controversial materials. Thus in order to discover more hateful speech, we researched online and find out a list contained both banned

and quarantined subreddits. Quarantined subreddits are subs that host no advertisement, and Reddit doesn't generate any revenue off toxic content. People can still acess those subs, but there will be a prompt warns telling people about the content on the sub. We have selected around 37 qurantined subreddit along with 10 normal subreddits.
By using the data ingestion pipeline, we have successfully extracted 5,000 posts from each of the 47 subreddits which is 235,000 posts in total. For each of the subreddit we have calculated **total_comment**: the total number of comments recieved for the posts contained in that subreddit, **avg_comment**: average number of comments received for the posts contained in that subreddit, **top_num_comment**: the maximum number of comments recieved by a post in that subreddit. The statistics for the top 5 subreddits that have the most total comments are shown in the table below. From the table, we can observe that the subreddit with higher number of total_comments also has higher number of average_comment. And we also want to figure out whether those hot subreddit also tend to contain more hateful speech.

subreddit	total_comment	avg_comment	top_num_comment
Politics (r/politics)	374,963	74	12,837
Pussy Pass Denied (r/pussypassdenied)	352,941	70	2,202
TumblrInAction: O Toucan, Where Art Thou? (r/TumblrInAction)	311,407	62	1,571
conspiracy (r/conspiracy)	223,516	44	949
KotakuInAction: The almost-official GamerGate subreddit! (r/KotakuInAction)	158,596	31	2,331

Thus we looked at the subreddit that has the most hateful posts. Below shows the top 5 subreddits.

subreddit	deleted	benign	hateful
Incest	1,067	3,295	609
Today I Fucked up(r/tifu)	771	3,708	487
TheRedPill	1,564	2,565	452
Jokes	766	3,758	405
Unpopularopinion	1,590	2,855	404

We then look at the labels at a higher level without group them into different subreddits. The table below shows the distribution of the labels among posts.

label	% post
deleted	10.8%
benign	84%
hateful	4.7%

Dig deeper into the content of the posts for different labeling groups, we investigate on the length of the content. From the table below, it shows that even though the min and max of the length of content in each

group is around the same, the average length of content for posts that are labled hateful is more than double of the average length of content for posts that are labled benign. Thus we can add this as one of our feature.

label	mean	min	max
benign	82.87	1	7549
hateful	176	1	7048

Another feture could be the number of comments under each post. The average length of comment for posts labeled hateful is relatively smaller than that for posts labeled as benign.

label	min	max	mean
benign	0	9,783	24
hateful	0	2,043	16

Moreover we also find difference in score for the two groups, the mean score of benign posts are generally higher than those of hateful posts.

label	mean_score
benign	32
hateful	11

Moreover, in order to evaluate the quality of the label, we have also done some textual analysis. We find out the top 30 words in posts after removing stop words for each of the groups. However, we have also removed about 20 words that appeared in both groups. Those should be the common words that appeared in the conversation and thus is not helpful as a feature for our classification.

malign_word	count	benign_word	count
fuck	5,835	amp	13,155
nigger	4,078	work	12,508
fucking	3,233	feel	12,388
shit	2,907	right	12,208
place	2,713	gt	11,256
sex	2,200	things	11,244
started	1,840	new	11,002
ass	1,800	need	10,629
went	1,717	years	10,374

Graph Data EDA

In order to maximize the model performance in later machine learning deployment, we restrict the type of users to active users only in our ingested data by removing users who never post, deactivated authors, auto-moderators and SnapshillBot. We also drop all posts authored by these removed users. With our cleaned data, we constructed heterogeneous graphs according to adjacency matrice definition.

We have 99,004 benign posts and 4,409 hateful posts in total after cleaning. Due to the imbalance between the spread of post labels, we will be weighing correspondingly in model constructions.

In our assumption, we hypothesize association between certain users who post hateful speech and the posts they interact with, thus we aim at investigating users' posting behaviors within subreddits.

We have 483,173 unique users in our data. 7% of users in our data have been involved with hateful posts. Among them, 44.26% of users have themselves create posts/comments labeled as hateful.

Percentage of users only post once	Proportion of users post only in 1 subreddit
44.95%	85.24%

We can observe that nearly half of the users post only once and are not active authors on Reddit. Most of them are only involved within one subreddit, thus their behavioral movements are representative of that subreddit.

In addition to general user, we also investigate hateful post users' behaviors specifically.

Proportion of users engaged in hateful post, only post once in a subreddit	Proportion of users engaged in hateful post, post only in 1 subreddit
20.01%	70.03%

We can observe that users who engage in hateful post are more active authors compare to general users.

Some users engage in both benign and hateful posts, and we found that among users who have engaged in hateful posts, 14% of there authored posts and comments are classified as hateful.

Proportion of speech for users engaged in hateful posts
28.15%

7. ML Deployment (new)

Metrics

Since we are performing binary classification, `True Positive, True Negative` plays a more crutial role in our classification model. Also, our label is extremely unblanced with few positive labels and much more negative labels. Becuse graph technique will be significantly influenced (invalided) by traditional balancing data technique like over-sampling and under-sampling, we will be evaluate our model with fowllowing metrics: `Recall, and Precision`. To catch more potential hateful posts, we favor Racall over Precision.

Baseline Model

Based on our tabular data EDA, we have determined a few features that maybe helpful in defining our baseline model, including number of comments, subreddit it belongs to, upvote score, length of the text body, and some

sensitive words that frequently appear in hateful posts shown in EDA. We use Logistic Regression, Random Forest, and Gradient Boost Classifier to train our models based on the features `['num_comments', 'subreddit', 'score', 'length', 'sensitive']` and classify whether a post is hateful.

- `num_comment`: number of comments for each post
- `subreddit`: the subreddit that the post belongs to
- `score`: the upvote score that the post receives
- `length`: the length of text body of the post
- `sensitive`: whether the post includes any of the sensitive words: 'fuck', 'nigger', 'shit', 'sex', 'ass'.

Hinreddit

Hinreddit will present methodologies over following graph techniques: `Node2vec`, `DGI`, and `NetMF`

Node2vec

The Node2vec model from the "node2vec:Scalable Feature Learning for Networks" paper where random walks of length `walk length` are sampled in a given graph, the embedding is learned by negative sampling optimization.

DGI

DGI, Deep Graph Infomax, model from the "Deep Graph Infomax" paper based on user-defined encoder and summary model $$\epsilon$$ and $$R$$ respectively, and a corruption function $$C$$

We use implementation from `pytorch_geometric` for our modeling to get the Graph embedding of latent features.

NetMF (not finished)

NetMF, Network Embedding as Matrix Factorization, method from the "Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec" paper, lays the theoretical foundation for skip-gram based network embedding methods, leading to a better understanding of latent network representation learning.

8. Experimental Result (new)

Baseline Model Result

Our current dataset is splited into 70% training and 30% test. We also adjust weights due to out data imabalance. The weights are inversely proportional to class frequencies, and our weights are as follows: 0.53 for benign posts and 9.30 for hateful posts. The resulting metrics for each classifier regarding the performance on the test set are listed in the table below.

	Precision	Recall	AUC
Logistic Regression	0.177778	0.723241	0.847873
Random Forest	0.334432	0.519403	0.780207
Gradient Boost Classifier	0.697987	0.221748	0.862229

From the table above, we can observe that the performances logistic regression produces relatively high recall, meaning that it identifies more hateful posts from all existing ones, while producing low precision, meaning that only a small portion of posts it identifies as hateful are truly hateful. On the other hand, gradient boost classifier produces opposite results with lower recall and higher precision. Both logistic regression and gradient boost classifier have higher AUC compared to random forest, meaning that they are both better at distinguish between hateful posts and benign posts.

Hinreddit Result

Node2vec

Small Data Result

Since the computational cost for Node2vec is large, and we have an overall large graph, we are going to limit our model to subreddit `incest`, which has 228 positive data and 1383 negative data (1611) in total.

	Precision	Recall	AUC
Logistic Regression	0.115044	0.302326	0.472591
Linear SVC	0.123894	0.325581	0.486005
Random Forest	0.000000	0.000000	0.500000

DGI

The result of implementing Heterogeous Deep Graph Infomax is presented below:

	Precision	Recall	AUC
Logistic Regression	0.083879	0.585419	0.644055
Linear SVC	0.089744	0.609358	0.660983
Random Forest	0.103790	0.262242	0.578475

NetMF

Not finished implementation

9. Discussion (new)

Result Analysis

As seen above, we have obtained fairly low precisions and recalls with our current user-post embeddings and models. The results can be understand together with our Exploratory Data Analysis. The data has shown that only 7% of users have ever engaged in hateful posts, and among them almost half of users have themselves write posts/comments that are labeled as hateful. Moreover, for users who have engaged in hateful posts, only around 28% of their posted speeches are labeled as hateful. These numbers suggest users have a small chance of creating their own hateful posts/comments although they have engaged in any of the hateful posts. Furthermore, even if they have created any, it is not a consistent behavior. This then demonstrates that our

initial hypothesis might not be accurate as mere relationships of users' reply behavior and authorship cannot provide much useful information in identifying hateful posts, which is then confirmed by our model results.

Due to the fact that our graph representation only embeds authorship and reply behavior, and as Node2vec and Deep Graph Infomax both are greedy in the training process, the models cannot clearly distinguish between hateful and benign posts, which is shown by the AUC values that are only slightly higher than, or even lower than, 0.5 and our baseline models that make use of post-related features.

Possible Improvement

First possible improvement can be done in the labeling process. To label our reddit data, we have trained a NLP classifier with labeled Wikipedia comments as well as pretrained Wikipedia vocabularies. It would be better if we can obtain labeled social platform data, such as labeled tweets to be trained with pretrained Twitter vocabularies provided by Glove.

Another possible improvement, which we originally would like to implement, is to include more user-to-user relations in our graph representations. Some examples include subreddit subsription lists and friends connection. We currently have a hard time including these relationships because user information features are still in development in the API we use, `pushift`. Although Reddit's own API, `PRAW`, offers related features, it unfortunately employs a different system of user ID from what pushift uses, and we are unable to connect them to obtain user information. This can be done as soon as `pushift` succesfully develops user information features.

Finally, we can also improve our algorithm to make use of the timeline data we obtain along with posts and comments. By adding a time feature, we can construct sequence nodes and feed in to Recurrent Neural Network models, such as Long Short-Term Memory.

10. Pipeline (updated)

ETL Process

target key: `data[-eda/test]`

- Create `config/data-params.json`, an example shown below. Information includes: POST_ARGS: parameter related to the post extraction part. META_ARGS: parameter related to the comment extraction part. The all the posts is sorted by the creation data and we extracted data prior to the date of `Tuesday,March 31 17:00:00 2020 PDT`.

```
{"POST_ARGS":
    {"sort_type":"created_utc",
    "sort":"dsc",
    "size":"1000",
    "start":"1585699200"},
"META_ARGS":
    {"filepath":".\/tests",
    "total":"1000",
    "meta":
["id","author","title","selftext","num_comments","created_utc","full_link"
,"subreddit","score"],
```

```
    "subreddits":
["amitheasshole","showerthoughts","politics","documentaries"]}}
```

- Sample a number of newest posts prior to a chosen daytime, the number specified in configuration file, from each subreddits specified in configuration file.
- Access and obtain reddit posts, reorganize, and same them as detailed in schema

Labeling Process

target key: `label[-test]`

- Train a multi-label classification model using `Keras` deep learning NLP model and kaggle dataset.
- Use the model to label post csv files stored in post path.
- Store the label csv files in label path.

Create Baseline Model

target key: `baseline[-test]`

- Extract simple features detailed in baseline section.
- Use the features to train linear regression, random forest, and gradient boost classifiers.
- Use the baseline models to predict labels and store the result in output directory.

Extracting Graph Data

target key: `graph[-test]`

- Extract Graph specified in Graph Extraction section.
- Combing post and user relationship and labeled data with `pandas`.

Graph Embedding

Node2vec Embedding

target key: `node2vec[-test]`

- create embedding from graph nodes into nodes' embedding vector for node2vec algorithm specified in Node2vec section.
- The parameter file is under `./config/node2vec.json`. With `-test` flag, the parameter file is under `./config/test-node2vec.json`.
- using Node2vec implementation in Pytorch Geometric to perform embedding (both random walk and word2vec) process.
- Parameters

```
{
  "BATCH_SIZE": 128,
  "EMBEDDING_DIM": 64,
  "WALK_LENGTH": 100,
  "CONTEXT_SIZE": 10,
  "WALKS_PER_NODE": 10,
  "LEARNING_RATE": 0.001,
  "NUM_EPOCH": 1,
  "P": 1,
  "Q": 1,
  "CUDA": true
}
```

`BATCH_SIZE`: the number of row to be trained in each batch.

`EMBEDDING_DIM`: the output embedding dimension.

`WALK_LENGTH`: the walk length of random walk.

`CONTEXT_SIZE`: the number of positive sample & negative sample in each walk.

`WALKS_PER_NODE`: the number of walks to be performed per node.

`LEARNING_RATE`: the learning rate of SGD optimization in loss function.

`NUM_EPOCH`: the number of epoch to perform embedding.

`P`: the parameter to control probability of walk to go back.

`Q`: the parameter to control probability of walk to go towards the end.

`CUDA`: True if use GPU for embedding.

DGI Embedding

target key: `infomax[-test]`

- create embedding from graph nodes into nodes' embedding vector for DGI algorithm specified in DGI section.
- the parameter file is under `./config/infomax.json`. With `-test` flag, the parameter file is under `./config/test-infomax.json`.
- using Node2vec implementation in Pytorch Geometric to perform embedding process.
- Parameters

```
{
"HIDDEN_CHANNELS": 64,
"SUMMARY": 64,
"LEARNING_RATE": 0.01,
"NUM_EPOCH": 300,
"CUDA": true
}
```

`HIDDEN_CHANNELS`: the hidden layer's dimension.

`SUMMARY`: the output embedding layer's dimension.

`LEARNING_RATE`: the learning rate of SGD optimization in loss function.

`NUM_EPOCH`: the number of epoch to perform embedding.

`CUDA`: True if use GPU for embedding.

11. Backlog (new)

Graph Extraction

We have tried different approaches upon graph extraction, as follow:

- Approach 1
  - Post - User (undirected) if Post contains User
  - User - User (undirected) if User replied by User
- Approach 2 (under file `graph_v1.py`)
  - Post - User (undirected) if Post contains User
  - User - User (directed) if User replied by User
- Approach 3 (under file `graph_v2.py1`)
  - Post - User_author (undirected) if Post written by User
  - User - User (directed) if User replied by User

Approach 1 was replaced by Approach 2 because it cannot present the relationship between User and User as a post-reply conversation.

Approach 2 was replaced by Approach 3 because it cannot present the author of the post as author is an important feature of user when the question comes to hateful post.

Approach 3 was replaced by our current approach since it breaks the connection between author and users who reply the post.

ML Deployment Backlog

Node2vec Backlog

Since our graph is extreamly large, we try to find a way to make the training process to be as optimized as possible (in terms of memory and time). We have done following approach w.r.p to Node2vec

- using self implemented random walk and word2vec in `gensim` with cutting users nodes
  - We try to cutting users nodes to lower dimension, but since the user nodes are much sparser than android API. We have to abandon this approach
- using implemented Node2vec in `pytorch geometric` library
  - Pytorch geometric can utilize GPU accerlerated API to expede our process of training, but it only support Node2vec random walk with parameter p=1 and q=1, which is basically a DeepWalk algorithm.

    - We try to implemente a custom random walk with torch tensor, but the lower abstraction of the random walk algorithm was not published, so we have to abandon pytorch geometric Node2vec.
- using implemented Random Walk in `Stellargraph` library
    - Random Walk in Stellargraph takes about 3 minutes to train a much smaller graph with only 1 subreddit . Since it does not have estimated time to finish random walk, we have not yet tested out in the large graph.

NetMF Backlog

We just found this paper two days before the deadline of this check point, and we are still in development of NetMF