What is the Content of a Dark Web Cyber Blackhat Forum?

CISC 499 Advanced Undergraduate Project
Queen's University
Supervisor: D. Skillicorn
Group: O. W., D. G., S. D.

Table of Contents:

Introduction:

Dark Web Cyber Blackhat forums are online groups where individuals with malicious intent can share information, tools, and techniques about cybercrime and hacking and facilitate illicit activity. Most of these forums are hidden from the public and often require special access; they are used by "black hat" hackers, who use their skill set for malicious purposes such as data theft, distributing malware, and phishing attacks. Discussions on Dark Web Cyber Blackhat Forums typically consist of participants sharing tips and tricks for hacking into systems, buying and selling stolen data, and organizing cybercrime activities.

In the initial stages of our project, we set out questions we wanted to find answers to in order to guide us in the right direction and display our findings in an effective way. From the starting question of "What is the content of Dark Web Blackhat Cyber Forums?", we further broke it down to "What do the participants talk about, and can we pick out malicious individuals?". If participants are sharing important information on the latest vulnerabilities, exploits, and malware, it is important that we find where these posts are coming from and which participants are contributing to the section. Next, we wanted to discover whether what is being said on these forums can be used to help predict future cyber-attacks. These forums typically share insights into current or planned cyber threats, so it is important for us to recognize whether

this can be used to anticipate cybercrime or malicious activities. Finally, we wanted to see if what we learned from these forums or discovered throughout our project could be brought to the attention of law enforcement. These forums can be the gateway to identifying potential malicious actors, so it is important that law enforcement be involved with these forums and the strategy by which we can analyze them to make them useful to us.

In this report, we will outline the use of data preprocessing and deep learning NLP models to analyze the content of online forums. Our initial questions serve as a guide for the project's objectives and learning process. We will discuss important aspects of our project, such as challenges faced and techniques employed, that can be utilized by law enforcement to detect and track malicious individuals.

BERTopic:

To deconstruct a forum with over 9 million posts, we relied on a Neural Network Python package called BERTopic. Written by Maarten Grootendorst, it primarily consists of two parts: BERT: Bidirectional Encoder Representation from Transformers, which converts text into a language computers can interpret and process efficiently, and c-TF-IDF: class-based term frequency–inverse document frequency, which performs topic modeling by processing the encodings through the TF-IDF algorithm and grouping them into ‘classes’. This allows us to process the dataset and compute visualizations of these topics, which gives us an overview of what is being discussed without necessarily needing to read every single post.

The raw data was provided to us by our professor. This leaves us with three steps: preparing the data, training the model, and analyzing the results in ways that answer our initial inquiries.

Preparing the data - csv_parse.ipynb:

The data begins as raw text. As we can see below, a scraped forum produces a lot of information.

intPost_PK	intCrawlID	txtLocation	txtJoinDate	dtJoinDate	txtGender	txtAuthor	txtAwards	txtPostDate	dtPostDate	txtReputation	txtRepPower	txtBody	txtRank	intPostNumber	intTotalPosts	txtProfile	txtRepliedTo	intTimesThanked	txtThanked	txtReligion	txtTitle	txtThreadIDP	dtDateTime	intThreadID	txtBody_Clean	intSubForumID	intForumID	intAuthorID	intInternalAuthorID	intGeneralSentiment	blnProcessed	intPost_PKModThree	intReplies	intReposts	intLikes	txtAuthorID_Text	txtExternalPostID
153972369	-1		May 19, 2011	5/19/2011 7:00:00 AM		JOSourcing		May 1, 2015	5/1/2015 12:00:00 AM			<article>	<blockquote class=messageText SelectQuoteContainer ugc baseHtml>	I don't see where artificial intelligence plays a role. Anyone care to point it out to an old bird?		</blockquote>	</article>		<div class=messageMeta ToggleTriggerAnchor>						8	125	Junior Member	3/28/2018 6:52:08 PM	757816	I don't see where artificial intelligence plays a role. Anyone care to point it out to an old bird?	252	157	187455	225511	0	0

For our purposes, the most important columns we will need are txtBody_Clean and txtAuthor. We extract these two columns using Pandas dataframes. Next, we need to parse out any unneeded artifacts that may not add to the legibility of our model. In our case, these are original posts that people would later quote when replying to original posters. We completed this in a JupyterLab file called csv_parse.ipynb. There are additional text preparation techniques presented at the bottom: stemming, removing symbols, and removing stopwords. We reverted our implementations that included these steps, as the model handled these cases automatically. Allowing the model to handle these cases creates a more coherent model with little additional overhead. Additionally, the reduced complexity of the corpus using these techniques did not make a difference in our use case as the model could not be trained in one shot anyway. Using simple techniques with dataframes to parse the data, we were able to reduce the number of characters by 22%. The resulting data - to contrast with the raw data of Forum157 - can be seen below after the execution of csv_parse.ipynb.

text	author
Cloaking is a search engine optimization technique in which the content presented to the search engine spider is different from that presented	Diamond Damien
Do the search engines allow this. If they discover what you are doing will they reduce your page ranking?	Timothy
Often when you get a high ranking page under quality keywords the first thing that will happen is your page gets stolen (called PageJacking). ...	twirly
There are 5 types of cloaking: * User Agent Cloaking (UA Cloaking) * IP Agent Cloaking (IP Cloaking) * IP and User Agent Cloaking ...	twirly
1) User Agent cloaking is good for taking care of specific agents. Wap, Wml pages for the cell phone crowd. Active X for the IE crowd.	twirly
With the assault of rogue spiders most sites are under, the growing trend of framing, agents that threaten your hrefs (smarttags), ...	twirly
and hundred of pages on the Cloaking issue....to be found on: http:dmoz.orgComputersInternetWeb_Design_and_Development ...	twirly
A lot of big website use cloaking too. I have heard Amazon.com uses cloaking.	Diamond Damien
I am currently using cloaking on a non essential domain and I am starting to see some un-godly spider activity. Below is the number of ...	Msnow
Sounds like cheating to me. Is it really necessary to cloak? I mean, with everybody trying to out-cloak each other, the results would be ...	spywarz
Is anyone still getting good rankings from black-hat cloaking?	elvis

Training the Model - onlinemodel_1000_3_3_10decay.ipynb:

With the data preprocessed, we were ready to train the model. This was the most difficult task of this project. The two checkpoints were fitting the amount of data we had to the amount of hardware available to us and tuning the hyperparameters.

Fitting the model on 9,500,000 posts was an enormous feat. We tried every performance improvement documented on the models github, such as using GPU-accelerated components and low memory settings, yet 256GB of RAM was not enough to Cluster the reduced embeddings in the UMAP stage of the fitting. This meant that we had to train the model incrementally with online learning, which is done by using algorithms that can be implemented incrementally. Our previous approaches, as well as our online learning outputs, are documented in the folder attached to this report.

Once we fitted a couple of working models, we were ready to adjust the hyperparameters to get a coherent picture of the forum. As discussed in more detail below, we adjust the parameters and compare the resulting topic hierarchy text tree from the model. For our final model, "onlinemodel_1000_3_3_10decay.ipynb" we found the following values to produce the most coherent output:

Online learning (uses different modules compared to the regular BERTopic package):

Number of Topics: 1000

- One could choose fewer topics, say around 250, although we worked with a higher number to have more depth when looking through the hierarchy tree of conversations. We can merge these topics later.

N-Gram Range: (3,3)

- A tuple of (3,3) means that the model only considers strings of exactly 3 words. For example: "I am selling". This was the perfect choice for our use case as it would accurately describe what individuals were talking about from a broad perspective. If we chose 1 or 2 as the minimum (the first number in the tuple), we would not be able to differentiate between topics that surrounded major subjects of one keyword, such as "money". Increasing the maximum to 4 or 5 would not give us much more information compared to 3, but would greatly scale the complexity.

Decay Rate: 10%

- A decay rate of 10% means that with every iteration in the training, 10% of the least important data would be truncated. This is necessary for creating topics, as it eliminates the outliers lying between classes. Keep in mind the number of iterations, as this will change the math behind tuning this hyperparameter. We only had 10 iterations.

Resulting model:

Running the online learning model

```
!TZ='America/New_York' date
```

```
Sun Mar 26 22:54:14 EDT 2023
```

```
from sklearn.cluster import MiniBatchKMeans
from sklearn.decomposition import IncrementalPCA
from bertopic.vectorizers import OnlineCountVectorizer

# Prepare sub-models that support online learning
umap_model = IncrementalPCA(n_components=5)
cluster_model = MiniBatchKMeans(n_clusters=1000, random_state=0,batch_size=16384)
vectorizer_model = OnlineCountVectorizer(stop_words="english", decay=.10,ngram_range=(3,3))
```

```
topic_model = BERTopic(umap_model=umap_model,
                       hdbscan_model=cluster_model,
                       vectorizer_model=vectorizer_model,verbose=True)
topics = []
iterations = 0
# Incrementally fit the topic model by training on 1000000 documents at a time
for docs in doc_chunks:
    print("iteration: "+str(iterations)+" of "+str(data_length // chunk_size)) #Estimating the total
    topic_model.partial_fit(docs)
    topics.extend(topic_model.topics_)

    iterations+=1

topic_model.topics_ = topics
```

Now that the model is trained, we can explore the content by printing the text tree. Using this technique, we have reduced more than 10 years of posts into 1000 topics and are free to explore the contents of the “blackhat” forum.

```
.
├─million links indexed_15 million links_seofbytiggpinterest services related_gscraper proxy services_
│    ├─million links indexed_15 million links_cheapest fastest captcha_captchatronixcom cheapest fastest_in
│    │    ├─million links indexed_15 million links_gscraper proxy services_services related scrapebox_seofbytigg
│    │    │    ├■─flat discount manual_backlinks 30 flat_comment backlinks 30_discount manual submission_related blog  ── Topic: 230
│    │    │    └─million links indexed_15 million links_services related scrapebox_gscraper proxy services_related sc
│    │    │         ├─million links indexed_15 million links_seofbytiggpinterest services related_related scrapebox gscrap
│    │    │         │    ├■─million links indexed_15 million links_seofbytiggpinterest services related_services related scrapeb ── Topic: 236
│    │    │         │    └■─million links indexed_15 million links_gscraper proxy services_related scrapebox gscraper_seofbytigg ── Topic: 192
│    │    │         └■─million links indexed_15 million links_links power 20_prdapa links power_seohigh prdapa links ── Topic: 158
│    │    └─indexificationcom 15 million_websites indexed captchatronixcom_cheapest fastest captcha_solving serv
│    │         ├─indexed dirty cheap_captchatronixcom cheapest fastest_indexificationcom 15 million_solving service i
│    │         │    ├■─million links indexed_15 million links_indexificationcom 15 million_instantlinkindexercom links webs ── Topic: 235
│    │         │    └■─indexed captchatronixcom cheapest_indexificationcom 15 million_cheapest fastest captcha_service inde ── Topic: 107
│    │         └■─million links indexed_15 million links_indexed dirty cheap_cheapest fastest captcha_solving service  ── Topic: 266
│    └─million links indexed_15 million links_related scrapebox gscraper_gscraper proxy services_seofbytigg
│         ├─million links indexed_15 million links_gscraper proxy services_seofbytiggpinterest services related_
│         │    ├■─million links indexed_15 million links_gscraper proxy services_seofbytiggpinterest services related_ ── Topic: 183
│         │    └─gscraper proxy services_related scrapebox gscraper_services related scrapebox_seofbytiggpinterest se
│         │         ├■─million links indexed_15 million links_related scrapebox gscraper_seofbytiggpinterest services relat ── Topic: 241
│         │         └■─services related scrapebox_gscraper proxy services_related scrapebox gscraper_seofbytiggpinterest se ── Topic: 167
│         └─million links indexed_15 million links_seofbytiggpinterest services related_related scrapebox gscrap
│              ├─million links indexed_15 million links_services related scrapebox_gscraper proxy services_related sc
│              │    ├■─million links indexed_15 million links_related scrapebox gscraper_services related scrapebox_seofbyt ── Topic: 262
│              │    └─million links indexed_15 million links_related scrapebox gscraper_seofbytiggpinterest services relat
│              │         ├─million links indexed_15 million links_services related scrapebox_gscraper proxy services_seofbytigg
│              │         │    ├■─million links indexed_15 million links_seofbytiggpinterest services related_services related scrapeb ──
Topic: 275
```

One thousand topics is great for understanding the content, but if we would like to extract other information from the corpus, we will need to combine branches in order to give us greater polarity between topics.

Professor Skillicorn's previous work with SVD and word2vec relied on querying a certain group of words to extract responses from certain groups of people in the forum. An example query of "not thanks" produces words that are usually associated with sharing content. If we clean the model by combining topics that are uninteresting from a security standpoint and leaving open branches that can tell apart different intentions, we are able to extract not just a response but even the group of posts that belong to that topic:

```
topic_model.find_topics("selling instagram bots")

([772, 702, 335, 943, 167],
 [0.6285648713436984,
  0.6253764478237855,
  0.605961329409267 1,
  0.5956234703899416,
  0.59096833090608])
```

The models associated with the analysis that follows for the rest of this report are found in bertopic_playground.ipynb, which contains the final model, its associated text tree, and any techniques used. The cleaned model with uninteresting topics combined and identified is saved in v3_cleaned_bertopic_playground.ipynb and loaded in with v4. Pictured below is the resultant text tree, showcasing topics related to bots.

```
└─dirt cheap prices_price lightning fast_pay monthly socialsboxcom_smm million dollar_dollar harvard l
    ├─bot facebook instagram_authority content great_google tumblr linkedin_tumblr linkedin star_turnaroun
    │    ├─bot facebook instagram_tumblr linkedin star_media bot facebook_authority content great_star authorit
    │    │    ├■─diversity websites authority_network 16 years_seo package powerful_redirects seo package_drops tf 20 ── Topic: 144
    │    │    └─bot facebook instagram_linkedin star authority_google tumblr linkedin_sherbme jarvee best_turnaround
    │    │         ├─bot facebook instagram_tumblr linkedin star_sherbme jarvee best_turnaround special deals_linkedin st
    │    │         │    ├─bot facebook instagram_star authority content_authority content great_sherbme jarvee best_google tum
    │    │         │    │    ├■─bot facebook instagram_sherbme jarvee best_turnaround special deals_star authority content_tumblr li ── To
    │    │         │    │    └■─bot facebook instagram_linkedin star authority_sherbme jarvee best_google tumblr linkedin_media bot  ── To
    │    │         │    └■─best social media_bot facebook instagram_linkedin star authority_authority content great_media bot f ── Topic:
    │    │         └■─niches accepted usual_accepted usual ones_usual ones accepted_ones accepted niches_accepted niches a ── Topic: 162
    │    └─google tumblr linkedin_star authority content_linkedin star authority_turnaround special deals_sherb
    │         ├■─turnaround special deals_tumblr linkedin star_authority content great_media bot facebook_star author ── Topic: 206
    │         └■─typespan attrtxtfollowing tag_instagram twitter pinterest_bot facebook instagram_sherbme jarvee best ── Topic: 184
    └─dirt cheap prices_prices smm million_socialsboxcom instant panel_harvard level authority_instant pan
```

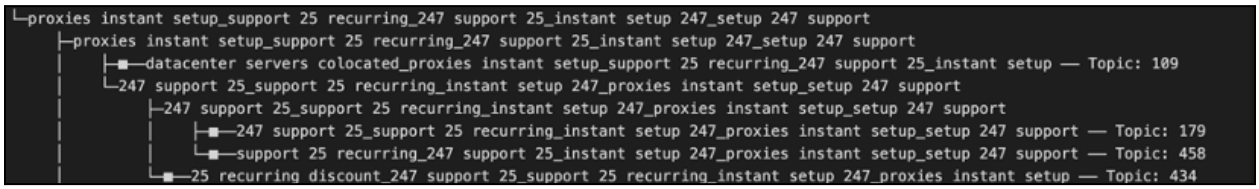
Analysis of Dark Web Cyber Blackhat Forums:

What do the participants talk about?

In the forum we were analyzing, we found about 99% of what participants were talking about to be uninteresting to us from a cybersecurity perspective. This forum focuses on search engine optimization. Some of the most discussed topics are link building, which is pointing other website pages to yours to rank higher, money making techniques, and ordering domains.

Many of these uninteresting posts consisted of participants thanking other users and other random discussions following pop culture such as “gucci gang”. From the interesting material, there were some noteworthy topics that could lead to interesting findings. There were a lot of discussions about bots. It seems participants were trying to figure out how to use bots to promote their personal websites. Below is a branch of our outputted hierarchy tree that points to topics talking about social media bots:

As a result, there were also topics on CAPTCHA avoidance to facilitate the use of their bots on social media platforms. In addition, there were a couple topics that discussed PayPal. They most likely used Paypal for users to pay participants for their service and tips. Finally, we found topics talking about Kaspersky, a Russian cybersecurity and antivirus provider. This can be a sign that participants were trying to evade Kaspersky antivirus. Another interesting topic included participants talking about proxy setups:



What can be learned from the forums to help protect against cyber attacks?

In terms of the result we found from the forum we analyzed, we can do very little to protect against cyberattacks. To find information to accurately predict and protect against cyberattacks, the forum posts need to be recent (within a day or week). Our forum posts were not within the past 3 years. That being said, using our strategy and project techniques, we could learn from any recent forums to protect against future cyber attacks and point to potential threatening individuals.

Can we pick out malicious individuals?

Yes we can using the following strategy:

Query for specific keywords in topics:

```
topic_model.find_topics("selling instagram bots")

([772, 702, 335, 943, 167],
 [0.6285648713436984,
  0.6253764478237855,
  0.6059613294092671,
  0.5956234703899416,
  0.59096833090608])
```

Get all the post related to the chosen topic:

```
# Get all posts for a specific topic number
single_topics_data = []
for index in docs_per_topics[772]: # Choose topic number here
    single_topics_data.append(data[index])
```

Get the authors name from suspicious posts:

```
search_string = single_topics_data[65] # Choose a post to search by
print(search_string)
```

```
***
```

```
print(df.loc[df["text"] == search_string,["author"]]) # Search the po
```

Then we can read the rest of their posts, and discover whether they are of any importance:

Get all of an authors posts

```
author_name = "GoodBook" # Get from previous cell
posts = df.loc[df['author'] == author_name, "text"] #select all posts with the above authors name

# Set the maximum column width to display (unlimited)
pd.options.display.max_colwidth = None
```

```
posts.shape # Number of posts
```

```
(216,)
```

Read his other posts

```
posts.head(30)
```

```
45294
That's an intelligent move by Google if it's true.
```

What can be brought to law enforcement's attention?

From the forum we were given, we cannot bring any specific topic to law enforcement that may be interesting to them in terms of specific malicious individuals and/or potential cyberattacks. But we can bring our techniques used in this project to law enforcement to show them how we analyzed these forums and how we can find the individuals responsible for a given post they may be interested in. Our techniques can be used by law enforcement on more recent forums to monitor them for potential security threats and find malicious participants.

Challenges:

A major challenge we faced in doing this project was having enough memory to train our model in one shot. Having 9 million posts to process and train was a tedious task throughout our project work. We had to consider things like: computing power; preprocessing the data to make it clearer; training the model many times to get coherent hyperparameters; and how to visualize the content once the model was trained. The Queen's School of Computing offered us a solid system, which initially we thought would be more than enough, but unfortunately it was not enough in our case. We had to train our model piecewise in order to fit all the posts and forum content. Training the model incrementally gives us fewer visualization options than if we could train it with all the data in one shot. In addition, changing either the parsed data or the hyperparameters of the model would make us have to retrain the entire model, making training the model and modifying it challenging and tedious to make incremental improvements.

References:

Alsadhan, N., Skillicorn, D., & Frank, R. (2017). (rep.). *Comparing SVD and word2vec for analysis of malware forum posts*. Kingston, Ontario.

Grootendorst, M. P. (2023). *Home*. BERTopic. Retrieved April 10, 2023, from https://maartengr.github.io/BERTopic/index.html

MaartenGr. (2023). *Maartengr/Bertopic: Leveraging Bert and C-tf-IDF to create easily interpretable topics.* GitHub. Retrieved April 10, 2023, from https://github.com/MaartenGr/BERTopic