Reading Thieves' Cant: Automatically Identifying and Understanding Dark Jargons from Cybercrime Marketplaces

Kan Yuan, Haoran Lu, Xiaojing Liao, and XiaoFeng Wang, ***Indiana University Bloomington***

https://www.usenix.org/conference/usenixsecurity18/presentation/yuan-kan

This paper is included in the Proceedings of the 27th USENIX Security Symposium.

August 15–17, 2018 • Baltimore, MD, USA

ISBN 978-1-931971-46-1

Open access to the Proceedings of the 27th USENIX Security Symposium is sponsored by USENIX.

Reading Thieves' Cant: Automatically Identifying and Understanding Dark Jargons from Cybercrime Marketplaces

Kan Yuan, Haoran Lu, Xiaojing Liao, XiaoFeng Wang
Indiana University Bloomington

Abstract

Underground communication is invaluable for understanding cybercrimes. However, it is often obfuscated by the extensive use of *dark jargons*, innocently-looking terms like "popcorn" that serves sinister purposes (buying/selling drug, seeking crimeware, etc.). Discovery and understanding of these jargons have so far relied on manual effort, which is error-prone and cannot catch up with the fast evolving underground ecosystem. In this paper, we present the first technique, called *Cantreader*, to *automatically* detect and understand dark jargon. Our approach employs a neural-network based embedding technique to analyze the semantics of words, detecting those whose contexts in legitimate documents are significantly different from those in underground communication. For this purpose, we enhance the existing word embedding model to support semantic comparison across good and bad corpora, which leads to the detection of dark jargons. To further understand them, our approach utilizes projection learning to identify a jargon's hypernym that sheds light on its true meaning when used in underground communication. Running Cantreader over one million traces collected from four underground forums, our approach automatically reported 3,462 dark jargons and their hypernyms, including 2,491 never known before. The study further reveals how these jargons are used (by 25% of the traces) and evolve and how they help cybercriminals communicate on legitimate forums.

1 Introduction

Underground forums are communication hubs for cybercriminals, helping them promote attack toolkits and services [25], coordinate their operations, exchange information and seek collaborations [24]. For example, Silk Road, a forum with estimated 30K-150K active users [30], served as a breeding ground for narcotics and other illegal drug businesses, leaving 214 communication traces every day. Such traces provide a deep insight into the ways cybercrimes are committed, criminals' strategies, capabilities, infrastructures and business models, and can even be used to predict their next moves. However, they are often written in "thieves' cant", using encoded words like popcorn (marijuana), cheese pizza (child pornography) to cover their meanings.

```
(1) My fav is slayers new rat, its open source,
    gonna have his rootkit implemented into it.

(2) Strains i manage these days are BLUEBERRY and
    NYC Diesel.

(3) I vouch for this user he crypted my athena
    code.
```

Figure 1: Example sentences with dark jargons, where dark jargons are highlighted in blue color. The first example exhibits jargon "rat" which means "remote access trojan"; The second example shows the jargon "blueberry" for "marijuana", while "athena" is the jargon for a kind of botnet framework in the third example.

Such *dark jargons* are often innocent-looking terms (e.g., popcorn), which are extensively used for online purchasing/selling drug, seeking cybercrime wares' developers, doxing Blackhat SEO techniques etc. Figure 1 presents some dark jargons and their semantics in the underground forums Silk Road and Darkode. Such deceptive content makes underground communication less conspicuous and difficult to detect, and in some cases, even allows the criminals to communicate through public forums (Section 6). Hence, automatic discovery and understanding of these dark jargons are highly valuable for understanding various cybercrime activities and mitigating the threats they pose.

Reading thieves' cant: challenges. With their pervasiveness in underground communication, dark jargons are surprisingly elusive and difficult to catch, due to their innocent-looking disguises and the ways they are used,

which can be grammatically similar to the normal usages of these terms (e.g., sell popcorns). Even more challenging is to discover their semantics – the underground meanings they are meant to hide. So far, most dark jargons have been collected and analyzed manually from various underground sources, an approach neither scalable nor reliable [19].

Some prior researches on cybercrimes report the finding of dark jargons, though these automatic or semi-automatic analyses approaches are *not* aimed at these underground cants. A prominent example is the study on *dark words*, the terms promoted by blackhat SEO [46]. The study introduces a technique that looks for the query words leading to the search results involving malicious or compromised domains, which however is not to detect dark jargons but blackhat SEO targeted *dark words* (see detail in Section 6, under "innocent-looking dark jargon"). Also, another prior study [31] shows that the names of illicit products (e.g., bot) automatically discovered from underground marketplaces include some jargons (e.g., "fud" which means "fully undetectable exploit"). None of these approaches, however, are designed to find dark jargons, not to mention automatically revealing their hidden meanings. Addressing these issues needs new techniques, which have not been studied before.

Cantreader. In our research, we propose a new technique, called *Cantreader*, that utilizes a new *neural language model* to capture the inconsistencies between a phrase's semantics during its legitimate use and in underground communication. Fundamentally, a dark jargon can only be captured by analyzing the semantic meaning using the context in which it appears. A key observation we utilize to automate such analysis is that an innocent-looking term covering dark semantics tends to appear in a totally different context during underground communication than when it is used normally. For example, on a dark market forum, "cheesepizza" is usually presented together with "shot", "photo", "nude" and others, while on other occasions, the term comes with "food", "restaurant", "papa johns" etc. Thus, the key of identifying the jargon in the cybercrime marketplaces is to find its semantic discrepancy with itself when used a legitimate reference corpus.

To this end, we need to address several technique challenges: (1) how to model a term's semantic discrepancy between two corpora? (2) how to handle the terms that have been used differently even in the legitimate corpora? (3) how to understand a dark jargon which is not explicitly explained in the communication traces? To address these issues, we enhanced the standard neural language model by doubling the number of its input layer's neurons to process the sentences from two different corpora. Such a neural network outputs two vectors for each input word, one for the good set and the other for the bad set, automatically making the semantic gap between the word's context measurable (Section 4.1). To control the false positives introduced by the variations in a term's legitimate use, our approach runs the new model to compare the semantics of the terms in the good set (legitimate communication traces) and their meanings in a reputable interpretative corpus (such as Wikipedia and dictionary). Any inconsistency detected here indicates that the term can have a large variation in its legitimate semantics (e.g., "damage" is a slang for game in some legitimate corpora, see detail in Section 4), and therefore should be filtered out to avoid false positives. Finally, to understand a discovered dark jargon, we propose a hypernym ("is-a" relation) based semantic interpretation technique, which uncovered a terms with "is-a" relation to the jargon (e.g., "popcorn" is a "drug").

We implemented Cantreader and evaluated its efficacy in our research (Section 5). Using four underground forum corpora and one corpus from a legitimate forum, our system automatically analyzed 117M terms and in the end, reported 3,462 dark jargons and their hypernyms. With its high precision (91%), Cantreader also achieved over 77% recall rate. Our code and the datasets are available at [26].

Discoveries. Running on over one million communication traces collected from four underground forums across eight years (2008-2016), Cantreader automatically identifies 3,462 dark jargons along with their hypernyms. By inspecting these dark jargons together with their hypernyms and the underground communication traces involving jargons, we are able to gain an unprecedented understanding of the characteristics of dark jargons, as well as their security implications. More specifically, we found that dark jargons are extremely prevalent in underground communication: 25% of the traces using at least one jargon. Interestingly, our research reveals the possible ways cybercriminals choose jargons: drug criminals tend to use fruit names for the drugs with different flavors such as "pineapple", "blueberry", "orange" and "lemon", while, hacking tool developers prefer mythological figures like "zeus", "loki" and "kraken". Also, given the long timespan of the corpora, we are able to observe the evolution of the jargons: e.g., roughly 28 drug jargons appear each month, with the increase rate of 5.2%.

In terms of their security implications, we were able to utilize the dark jargons to discover and analyze criminal communication on *public* forums. Particularly, we detected 675 such traces on Reddit, the largest US forum. These criminal traces are related to various criminal activities such as illicit drug trade or sharing the "drug trip" experience. Also interestingly, using dark jargons, we even found 478 *black words*, which are another criminal

cant that unlike innocent-looking jargons, barely appear in legitimate communication: e.g., "chocolope" for marijuana, 'li0n" for crypter and "Illusi0n" for trojan.

Contributions. The contributions of the paper are as follows:

• *Novel dark jargon discovery technique*. We present Cantreader, the first fully-automated technique for dark jargon discovery and interpretation, which addresses the challenge in effective analysis of massive cybercriminial communication traces. Our approach leverages a new neural language model to compare the semantics of the same term in different corpora and further identify their hypernyms. Our evaluation shows that Cantreader can effectively recover and interpret dark jargons from underground forum traces, which cannot be done by any existing techniques, to the best of our knowledge.

• *New findings*. Running Cantreader on 375,993 communication traces collected from four underground forums across eight years, our study sheds new light on the characteristics of dark jargon, and the possible implications that they may have on criminal traces recognition and black word identification. The new understandings are invaluable for threat intelligence gathering and analysis, contributing to the better understanding of threat landscape.

2 Background

2.1 Cybercrime Communication

Underground forum. As mentioned earlier, the underground forum is an important component of the cybercrime ecosystem, a critical communication channel for coordination of malicious activities and doing underground business. These forums are known to host some of the world's most infamous cybercriminials. For example, the members of the "Lizard Squad" group were active members of Darkode [4], multiple drug dealers sold drug through Silk Road on a large-scale[17]. Hence, communication traces in the underground forum are considered to be an important source of cyber threat intelligence gathering. The rich information disclosed by such communication sheds light on the adversary's strategy, tactics and techniques, and provides the landscape of the fast-evolving cybercrime.

In our research, we studied the communication that took place on four infamous underground forums: Darkode (sale and trade of hacking services and tools), Hack Forums (blackhat hacking activities discussion), Nulled (data stealing tool and service) and Silk Road (illegal drug), including 375,993 traces (i.e., threads of posts) from 03/2008 to 05/2016 (4132 per month). In addition, we observe the number of the communication traces increases rapidly in all underground forums, which makes manual semantic analysis increasingly difficult.

Dark jargon. In our research, we consider a dark jargon to be an innocent-looking term used in the criminal community to cover its crime-related meaning. Such jargons often represent illicit goods, services, criminal tactics, etc., for the purpose of evading the law enforcement's detection. For example, drug traffickers have a long history to use dark jargon to describe illegal drugs to confuse eavesdropping federal agents. These jargons serve as a barrier for the "outsider" to understand criminals' conversations. Hence, identifying and understanding them are considered as a critical task for fighting against cybercrimes. For example, to better understand the drug trade business, Drug Enforcement Administration (DEA) intelligence program compiled a set of dark jargons to decipher forensic data and evidence or information gathered like traffickers' receipts.

Due to the dynamic and fast-evolving nature of cybercrimes, the vocabulary of dark jargons continues to change, adding new terms and dropping old ones. Also, every subgroup of criminals such as drug traffickers create their own jargons. Hence, it is important to continuously discover and interpret new jargons, timely updating the vocabulary list. This is by no means trivial. As an example, the drug jargons released by DEA have been complained to be misinformed and decades behind the time [19]. Considering the huge amount of underground communication traces collected, new techniques need to be developed to automate dark jargon identification and understanding.

2.2 Neural Language Model

Neural language model has been found very efficient for learning high-quality distributed representations of words (word embedding), which capture a large number of precise syntactic and semantic word relationships [28, 36, 37]. It aims at finding a parameterized function mapping a given word to a high-dimensional vector (200 to 500 dimensions), e.g., $v_{man} = (0.2, -0.4, 0.7, ...)$, that represents the word's relations with other words. Such a mapping can be learned with different neural network architectures, e.g., using the continual bag-of-words model and skip-gram, to analyze the contexts of input words from a large corpus [37]. Such a vector representation ensures that syntactically or semantically related words are given similar vectors, while unrelated words are mapped to dissimilar ones.

Architecture. Given the training set of a neural language model represented by a sequence $w_1, w_2, ..w_{|V|}$ of words, the objective is to learn a "good model" $f(w_1, ..w_{|V|}) =$

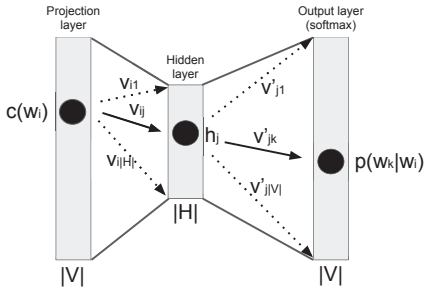


Figure 2: An example of neural language model.

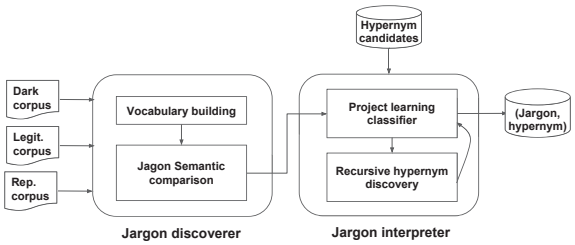


Figure 3: Overview of the Cantreader infrastructure.

$\prod_{t=1}^{|V|} \prod_{-m \le j \le m} P(w_{t+j}|w_t)$ where m is the size of training context, in a sense that it produces the highest likelihood for observing the context words $\{w_{t+j}, -m \le j \le m\}$, given the target word w_t and the training set, in terms of a softmax function $\frac{e^x}{\sum e^x}$ [29].

Neural language model architectures are essentially feed-forward networks, usually but not necessarily limited to only a single hidden layer. Figure 2 shows a typical neural network with one hidden layer used by the skip-gram model, an instance of neural language model allowing fast training. The input layer consists of $|V|$ neurons accepting a word w_i as a one-hot vector $c(w_i)$. For a word vector with $|H|$ features, the hidden layer with $|H|$ neurons takes the vector as the input, and output a $|H|$-dimension word vector. The output layer is a softmax regression classifier with $|V|$ neurons. Specifically, each output neuron has a weight vector that multiplies with the word vector from the hidden layer, before applying the softmax function to the result.

Sub-sampling. In large corpora, the most frequent words (e.g., "in", "the", "a") usually provide less information than rare words. For example, observing the co-occurrence of "woman" and "queen" is more valuable than seeing the co-occurrences of "queen" and "a". To counter the imbalance between the rare and frequent words, some neural language models such as the skip-gram model apply a simple subsampling approach: each word w_i in the training set is discarded with a probability determined as by computing $P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}}$ where $f(w_i)$ is the frequency of the word w_i and t is a chosen threshold, typically around 10^{-5}. The formula here aggressively subsamples words whose frequency is greater than t, while preserving the ranking of the frequencies. It accelerates learning and significantly improves the accuracy of the learned vectors of the rare words.

Word vector properties. Interestingly, the vector representations of neural language model capture the syntactic/semantic relations between words: e.g., the vectors for the words 'queen', 'king', 'man' and 'woman' have the following relation: $v_{queen} - v_{king} \approx v_{man} - v_{woman}$. Also, the same property also applies to hypernym-hyponym relations. For example, $v_{womon} - v_{queen} \approx v_{man} - v_{king}$ where 'woman' is the hypernym of the hyponym 'queen' and 'man' is the hypernym of the hyponym 'king'.

2.3 Hypernym identification

Hypernym Identification is an NLP technique to identify a generic term (hypernym) with a broad meaning that more specific instance (hyponym) falls under. For example, "woman" is a hypernym of "queen". Hypernym describes an important lexical-semantic relation and information abut it helps understand the semantics of its instance: e.g., knowing that "Tom Cruise" (hyponym) is an "actor" (hypernym) helps a question answering system answer the question "which actors are involved in Scientology?". Hypernym identification can be addressed by either distributional or path based approaches. The former [41, 32] uses distributional representations (such as word embedding) of the terms for hypernym identification. The latter [43, 44] leverages the lexico-syntactic path connecting the terms to detect hypernym. In our research, we utilize the distributional based method to find out the hypernym of a jargon so as to understand its semantics. This is because the path based methods require corpora following strict grammar structure, and also the hypernym and hyponym terms should occur together in the corpus, which is often not the case for underground forum data.

3 Cantreader: Overview

Cybercriminals on underground forums often pick common, innocent-looking words as their jargons to obfuscate their illegal communication. Identifying such dark jargons and discovering their semantic meaning, is difficult due to the stealthy nature of dark jargons. However, regardless of what a word looks like, its true semantics can be observed from its context. For example, when the "popcorn" means a snack, it often comes with "eat" or

"chocolate", while when it refers to marijuana, "nugz", "buds" and others would show up around the word.

This observation is the key to the automated discovery and understanding of dark jargons and is fully leveraged in the design of Cantreader. Our approach utilizes a novel neural language model to learn a word's semantics from its context during legitimate conversations and in underground communication respectively, and then compares the semantics to identify the consistency that indicates its potential use as a dark jargon. For each discovered jargon, we further perform a hypernymy-based semantic analysis to discover its underground meaning. Below we present the high-level design of this technique and explicate how it works through an example.

Architecture. Figure 3 illustrates the architecture of Cantreader, which includes two components: *the discoverer* and *the interpreter*. The discoverer analyzes the words included in an underground communication corpus and compares their semantics with that learned from legitimate corpora to identify dark jargons. More specifically, the component first filters out the words from the dark corpus, whose semantics is either insignificant or unlikely to be accurately learned with the neural language model. It then applies the Semantics Comparison Model (Section 4.1) to calculate two semantic similarities, $Sim_{dark,legit}$ and $Sim_{legit,rep}$, for each input word: the former between a dark forum corpus and a legitimate forum corpus, and the later between the legitimate forum corpus and a reputable interpretative corpus. A word is reported as a jargon only if $Sim_{dark,legit}$ is small and $Sim_{legit,rep}$ is large. The interpreter, on the other hand, uses a learning-based automatic hypernym discovery technology to interpret dark jargons by finding their hypernyms. From the public ontology (e.g., Wikidata), we collect a set of hypernym candidates of interest. The interpreter can predict whether any of them is actually a hypernym of a dark jargon.

An example. We take "popcorn" as an example, which normally, means a snack, but is also used as a slang for marijuana on the underground market such as Silk Road. Here we use Silk Road as the dark corpus (C_{dark}), Reddit as the legitimate corpus (C_{legit}), and English Wikipedia as the reputable interpretative corpus (C_{rep}), to demonstrate how Cantreader could discover and interpret the jargon.

After preprocessing all the corpora, Cantreader first trains two Semantic Comparison Models on (C_{dark}, C_{legit}), and (C_{legit}, C_{rep}) respectively. Both models output a pair of word vectors for "popcorn". The similarity between these two vectors (cosine similarity in our research) describes the similarity of the word's semantics in the two corpora. For "popcorn", we have $Sim_{dark,legit} = 0.256$ and $Sim_{legit,rep} = 0.474$. This indicates that "popcorn" carries very different meanings in the dark and legit corpora, but more similar ones across the legit and reputable corpora. So, it is labeled dark jargon by the discoverer. To find out the dark semantics of the word, we leverage an public ontology [20] including the terms of cybercriminal activities and illegal products exchanged on underground markets (such as RAT and marijuana), and a projection learning model to determine whether the word has an "is-a" relation with a class under the ontology. In the example, our model reports a probability of 93% that "popcorn" is a jargon for "marijuana".

4 Design and Implementation

4.1 Semantic Comparison Model

Fundamentally, Cantreader's jargon identification procedure is based on the fact that a word covering dark semantics tends to appear in a totally different context during underground communication than when it is used normally. In order to uncover such a difference, we propose our *Semantic Comparison Model*, which extends the neural network (NN) architecture of Word2Vec [36, 37] to analyze and compare the contexts for a given term.

Word2Vec model. Word2Vec (Figure 2) is a neural word embedding approach that uses shallow, two-layer neural network to learn a statistical language model (e.g. skip-gram model) from a large corpus. The NN applies one-hot encoding at the input layer, and identity activation function at its hidden layer. As an unsupervised learning model, when the training is done, Word2Vec outputs the weights of the hidden layer M in the form of a $|V| \times |H|$ matrix, where $|V|$ is the size of the input vocabulary and $|H|$ is the size of hidden layer. Consider $M = [v_1, v_2, ..., v_{|V|}]^T$. For a word w_i (whose one-hot vector has 1 at its i-th entry), the model assigns the i-th row in M, v_i, as the word's the embedded vector. Thus, Word2Vec maps words to vectors in $|H|$ dimension space.

The intuition behind Word2Vec is that if two different words have similar contexts in a corpus, then given the contexts, the NN is supposed to make similar predictions for these two words. Hence the training process will learn the weights to produce similar hidden layer outputs for these two words. Since the NN applies the identity activation function at hidden layer, the hidden layer output of the word w_i is exactly v_i, i.e. the embedded vector of that word. Therefore, embedded vectors are justified representations of the contexts of words, which, in turn, represent the semantics of words. Also the similarity between these vectors describes the similarity between the semantics of these words (see Section 2).

Our model. Word2Vec can be very useful if we want to find the semantic similarity of different words whose vector representations are trained over *the same corpus*. However, for dark jargon detection, we need to compare the semantics of the same word across different corpora, e.g., one for legitimate conversation and the other for underground communication. This cannot be done by simply combining these corpora together, which loses a word's context information in individual corpora. Nor can we train two separate models on the two different corpora, since the relations between the input layer and the hidden layer are nondeterministic, due to the random initial state for the Word2Vec NN, and the randomness introduced during sub-sampling and negative sampling (Section 2.2). As a result, for a given word, the NN produces different vectors each time when it is trained on the same corpus, which makes cross-model semantic comparison meaningless.

So the key challenge here is how to make a word's vectors trained from different corpus comparable. To address this challenge, we designed Semantic Comparison Model (SCM), a new network architecture based on Word2Vec NN, which doubles the size of the input layer without expanding either the hidden or the output layer. The idea is to let the same word from two different corpora to build their *separate* relations, in terms of weights, from the input to the hidden layer during the training, based upon their respective datasets, while ensure that the contexts of the word in both corpora are *combined* and *jointly* contribute to the output of the NN through the hidden layer. In this way, every word has *two* vectors, each describing the word's relations with other words in one corpus. In the meantime, these two vectors are still *comparable*, because they are used together in the NN to train a *single* skip-gram model for predicting the surrounding windows of context words.

To formally describe SCM, we first define an extended one-hot encoding:

$$e(w) = \begin{cases} [v_{zeros}, v_{onehot}(w)] & \text{if } w \text{ is from corpus}_1 \\ [v_{onehot}(w), v_{zeros}] & \text{if } w \text{ is from corpus}_2 \end{cases} \quad (1)$$

where v_{zeros} is an all-zero vector of $|V|$ dimensions, and $v_{onehot}(w)$ is the standard one-hot vector of word w in the input vocabulary. This encoding function converts words from two corpora to one-hot vectors of $2|V|$ dimensions, which enables SCM to get input from two corpora during the training stage. It also gives different distributed representation for the same word from different corpora, which ensures the two corpora to be treated differently by the model.

Since we double the size of the input layer, the weights of the connections between the input and the hidden layer M can now be represented as a $2|V| \times |H|$ matrix. We split it into 2 $|V| \times |H|$ matrices, $M = [M_1, M_2]$, where $M_1 = [v_{1,1}, v_{2,1}, ..., v_{|V|,1}]^T$ and $M_2 = [v_{1,2}, v_{2,2}, ..., v_{|V|,2}]^T$. As we can see here, for each word i, SCM outputs a pair of $|H|$-dimensional vectors: $v_{i,1}$ learned from corpus$_1$ and $v_{i,2}$ from corpus$_2$. The word's cross-corpora similarity can be measured by the similarity of these two vectors.

Table 1: Training settings

parameter	value	parameter	value
language model	skip-gram	minimal word occurrence	10
hidden layer size	200	hierarchical softmax	off
window size	10	sub-sampling	1e-4
negative sampling	25	iterations	30

Model effectiveness analysis. Our new architecture fully preserves the property of the Word2Vec model, in terms of comparing the semantic similarity between two words. Consider any two words from the corpora, no matter whether they come from the same dataset or not, if they are similar semantically, they should have similar contexts, that is, similar co-occurred words in the corpora. As a consequence, the NN should generate similar outputs for the two words. The output of the NN is determined by the output of the hidden layer and their connections with the hidden layer nodes, in terms of weights. Since the same set of output-layer weights is shared by all input word, similar NN outputs lead to similar word vectors. In the meantime, unlike Word2Vec, SCM uses two different corpora but learns every word's context from just one of them. So a word may have two contexts, one from each corpus. This property preserves a word's semantics in different scenarios (legitimate interactions vs. underground communication), which is critical for detecting dark jargons.

To analyze this architecture, we ran SCM on the *Text8* corpus [1], which is a 100MB subset of Wikipedia. The experiment settings is described in Table 1 and results are elaborated below.

Experiment 1. In the experiment, we used Text8 as both input corpora for our SCM. For each word in the vocabulary, the model generated a pair of vectors, each representing its semantics in the corresponding corpus. Since the two input corpora here are identical, the cosine similarity of every vector pair should all be close to 1, if SCM can capture the words' semantics in both corpora correctly. Our experiment shows that for every word in the corpora, the average cosine similarity between its two vectors is 0.98, with a standard deviation 0.006.

As a reference, we trained a Word2Vec model on the same corpus *twice*, and calculated the cosine similarities between the vectors of the same words. Here the average similarity is 0.49 and standard deviation 0.078, indicating that the vectors from the two models cannot be com-

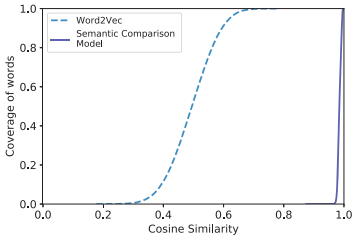


Figure 4: Results of experiment 1 in CDF.

Table 2: Results of Experiment 2

replacing word pair	similarity
(chemist → archie)	0.65
(ft → proton)	0.56
(universe → wealth)	0.67
(educational → makeup)	0.66
(nm → famicom)	0.45

pared, due to the training randomness. Figure 4 presents the cumulative distribution of the results.

Experiment 2. We then looked into SCM's capability to capture a word's cross-corpus semantic difference, which is at the center of jargon discovery. To this end, We randomly chose 5 words from the Text8 corpus and replaced them with 5 other words (see Table 2) to construct a new corpus Text8$_{syn}$. In this way, these replacements become "jargons" of the original words in the new corpus Text8$_{syn}$. Then we trained our architecture on Text8 and Text8$_{syn}$, which took in both corpora to learn a SCM. From the new model, again we compared the similarity between each word's two vectors (one from Text8 and the other from Text8$_{syn}$). The results are presented in Table 2. Since the replacing word's contexts (e.g., archie) in Text8$_{syn}$ became different to those in Text8 as recorded in replaced trace rate, all the replaced words were found to have small similarities in two corpora: the average similarity is 0.98 with a standard deviation of 0.01. This experiment shows that our SCM is able to capture a word's cross-corpora semantic difference.

Experiment 3. Finally, we want to measure the quality of the word vectors generated by SCM. For this purpose, we utilized the code and the test set provided by Tomas Mikolov [22] for evaluating the quality of word vectors. The test set includes a list of syntactic and semantic relations (such as capital of the country, adjective-adverb, etc.), and a number of test cases (such as Athens-Greece, Baghdad-Iraq) under each such a relation. The quality of word vectors is determined by semantic relations among these vectors: e.g., $v_{Athens} - v_{Greece} + v_{Iraq}$ is supposed to result in a vector very close to $v_{Baghdad}$.

In this experiment, we trained an SCM using Text8 along with a snapshot of *Nulled* [12], a collection of communication traces from an underground forum. On the word vectors produced by the model, we ran Tomas Mikolov's code to evaluate their qualities. The idea is to compare the vectors related to the Type8 corpus with those produced by the Word2Vec model trained over the same corpus. The experiment demonstrates that indeed the quality of the SCM vector (an accuracy of 46%) is in line with those generated by Word2Vec (50%), indicating that the benefit of semantic comparison across corpora does not come with the cost of vector quality.

4.2 Jargon Discovery

At the high-level, the discoverer is designed to find a word that tend to appear in different contexts on a dark forum than on a legitimate one. Such a semantic inconsistency can be captured by SCM.

Specifically, the discoverer takes a dark forum corpus (C_{dark}), a legitimate forum corpus (C_{legit}), and a reputable interpretative corpus (C_{rep}) as its input. After preprocessing these input corpora to build a shared vocabulary, it computes the cross-corpus similarities for each word by training two SCMs, one on C_{dark} and C_{legit}, and the other on C_{legit} and C_{rep}. After filtering out the words with special meanings in C_{legit}, our approach detects jargons whose similarities are low in the first model and high in the second. Here we elaborate on these individual steps.

Vocabulary building. Bootstrapping the whole discovery process is the generation of a vocabulary from the three input corpora. The vocabulary of SCM is the input word set for the model, including all "*words of interest*" in the intersection of good and bad corpora $C_{dark} \bigcap C_{legit}$, which we will explain later. Every word in the vocabulary corresponds to a specific dimension on the one-hot vector V. As mentioned earlier, the whole input of an SCM is two such vectors, one for each corpus (i.e., C_{legit} and C_{dark}, or C_{legit} and C_{rep}).

From the dark corpus, the "words of interest" are chosen by dropping all the "non-interesting" words. Specifically, we first filter out all stop words (common words like "the", "on", etc.), since their semantics is insignificant for finding jargons. In our research, these words are identified using NLTK [11]'s English stop words list.

Also importantly, we need to remove the words whose semantics cannot be effectively learned from the corpora. Particularly, the embedding techniques like Word2Vec and SCM all rely on a word's context to deduce its semantics and embed it into a vector space. If such contexts are not sufficiently diverse in a corpus, the embedded vector becomes biased and specific to the corpus.

Standard Word2Vec implementation uses a parameter `min_count` for this purpose. Those words whose occurrences in a corpus go below that parameter are excluded since they may not be effectively learned from the infor-

mation provided by the corpus. This approach, however, does not work well for our purpose: on forums, people tend to quote each other's posts, repost, and copy-paste published content to their own text. As a result, the same piece of text may appear on a forum repeatedly, and the words involved, though may occur across the corpus for many times, are always under the same context, whose semantics therefore cannot be effectively learned.

To address this issue, we introduce a new metric, called *windowed context*, to measure a word's context diversity. Given a window size k, the windowed context of a word w is a contiguous sequence of words start at k words before w and ends at k words after. For example, in the sentence "The quick brown fox jumps over the lazy dog", with a window size 2, the windowed context of word "fox" is ("quick", "brown", "fox", "jumps", "over"). Using this metric, we measure a word's diversity based upon its number of unique windowed context (`num_wc`) in a corpus. Those with a diversity below a given threshold in either C_{dark} or C_{legit} are removed from our vocabulary. In our research, we set the window size to 5 for the discoverer and the threshold to 20.

Jargon semantics comparison. After the vocabulary is built, the discoverer trains an SCM using the targeted dark corpus C_{dark} and a reference corpus C_{legit}. The purpose is to compare every word's two embedded vectors (one for each corpus) by calculating their cosine similarity $Sim_{dark,legit}$, for the sake of identifying those with discrepant meanings across the corpora.

However, just because a word has different semantics across the dark forum and the legitimate corpus does not always mean that it is a dark jargon. Particularly, if the legitimate corpus includes formal documents such as those from Wikipedia and news articles, false positives can be introduced. This is because words are used differently in these documents than in less formal forum posts. For example, on forums, "man" is commonly used as an expression of greeting, or an interjection to express anger or displeasure, while in a more formal context, it usually means an adult male. Another example is "peace", which is frequently used as a way to say goodbye in the forum language. To avoid misidentifying those "forum terms" as jargons, Canreader utilizes posts on a legitimate forum as the reference C_{legit}. Specifically in our research, the legitimate data were collected from `reddit.com`.

Unique context. Using legitimate forums as the reference corpus, we can avoid most false positives introduced by the semantic comparison. However, still we cannot eliminate the situations where some less harmful terms are treated as dark jargons, due to their unique contexts in the legitimate corpus, which can be different from their generic semantics actually used on dark forums. For example, we found that the word "damage" on `reddit.com` often appear during the discussion of computer games and as a result, its context becomes very much biased towards settings in the games (such as "heal", "stun" and "dps"); on Silk Road, however, "damage" preserves its original meaning.

To filter out the terms unique to the good set (C_{legit}), the discoverer compares every vocabulary word in the set to the same word in another legitimate corpus, in terms of their semantics. The new corpus, which we call *reputable set* C_{rep}, is supposed to include more formally-written documents that largely use each word's dictionary meaning. In our implementation, we chose Wikipedia as C_{rep}. Training an SCM on C_{legit} and C_{rep}, the discoverer is able to compare each word's semantics on both corpora ($Sim_{dark,legit}$) to detect and remove those carrying unorthodox contexts in the good set.

Threshold selection. As mentioned earlier, the discoverer reports a word as a dark jargon if its semantic similarity in C_{dark} and C_{legit} is below a threshold (different meanings across good and bad sets), while its similarity in C_{legit} and C_{rep} is above the threshold (similar meaning in two good sets). The challenge is, however, how to determine the threshold, which turns out to be non-trivial. In our research, we found that SCM tends to give larger cosine similarities to the words with diverse semantics. This is because a word with more meanings usually covers more different words in its context, so for such a word, its contexts in one corpus tend to have a larger overlap with that in another. Hence, we need different thresholds: a larger one for those with diverse semantics, and a smaller one for those with fewer meanings.

For this purpose, the discoverer first groups all vocabulary words into different classes according to their semantic diversities, as estimated using the numbers of *synsets* in WordNet [23]. Our implementation defines 4 classes: words having 0 *synsets* (not covered in the *wordnet*), between 1 to 4, between 5 to 8, and larger than 8. Over the classes, our approach runs a statistical outlier detection based on z-score [29] to find the thresholds. In our research, we use $z = 1.65$, so for each class, the discoverer simply computes the mean μ and standard deviation σ for the cosine similarities of the vector pairs as produced by an SCM for individual words and set $\mu - 1.65\sigma$ as the threshold for that model. Assuming in each class, the similarities follow a Gaussian distribution, the threshold we selected ensures that a normal sample (a word with similar semantics in two corpora) has only 5% chance to go below the threshold, in terms of the cosine similarity between its two embedded vectors.

4.3 Jargon Understanding

Once a possible jargon has been discovered, Cantreader runs the interpreter to make sense of it. Finding the pre-

```
SELECT ?x ?xLabel WHERE {
  SERVICE wikibase:label {
        bd:serviceParam wikibase:language
                "[AUTO_LANGUAGE],en".
  }
  ?x wdt:P279 wd:Q7397.
}
LIMIT 100
```

Figure 5: SPARQL example

Table 3: Summary of the corpora

corpus	# traces	# unique words	words per trace	timespan
Silk Road	195,403	1,183,506	1,321	6/2011 - 11/2013
Darkode	7,418	20,036	419	3/2008 - 3/2013
Hack Forum	52,654	30,020	211	5/2008 - 3/2015
Nulled	120,518	264,173	484	11/2012 - 5/2016
Reddit	1,190,346	3,497,646	1,190	-
Wiki	249,336	9,045,012	557	-

cise meaning of a dark jargon is challenging, due to lack of enough information to differentiate the contexts of related terms, particularly with the succinct expressions typically used on forums. On the other hand, we found that it is possible to gain some level of understanding of a jargon, by classifying it to a certain category under a specific hypernym. For example, though we may not have enough information to interpret "horse" as heroin, we may still be able to determine that it is a kind of illegal drug. Such understanding can also help a human analyst quickly decrypt the term, to find out its exact semantics.

This understanding is *automatically* generated by the interpreter as follows. First, it produces a set of hypernym candidates from the common products people trade on underground forums. Then our approach analyzes the semantics (in terms of embedded vectors) of a given jargon and all the candidates, running a classifier to find out whether any of them is a hypernym of the jargon.

Hypernym candidates generation. Our interpreter automatically expands a set of *seeds* to find hypernym candidates. These seeds are picked manually, including a small set of product categories discovered from underground forums as illustrated in Table 6. Using the seeds, the interpreter discovers other hypernym candidates by querying the *Wikidata* [20] database for the subclasses of the entities in the seed set.

Wikidata is a free and open knowledge base. It provides the Wikidata Query Service [21] that enables users to query its ontology using the SPARQL language [18]. Figure 5 shows the example to search for the subclasses under "software", where `wdt:P279` represents the *subclass of* relation, and `wd:Q7397` describes the *software* entity.

For each category (e.g., drug) in the seed set, we use Wikidata to find all its direct and indirect subclasses, and generate a tree rooted at the category in the seed. In this way, our approach generates a forest (a set of trees) out of the seeds where each node is a hypernym candidate.

Projection learning. Prior research demonstrates that an effective way to find hypernym relations is using a model learned from the semantic links between words, using the embedding techniques [32]. In our research, we follow the similar idea to build our interpreter. Specifically, for a given jargon, our approach uses its embedded vector together with a hypernym candidate's vector (from the same SCM) as a feature to determine the probability that they indeed have the hypernym relation. For this purpose, we adopt a binary random forest classifier, which unlike the multi-output linear regression model used in the prior research [32], can leverage the information not only from positive but also from negative samples to identify the decision boundary. This classifier was trained in our research using our hypernymy dataset described in Section 5.1.

Recursive hypernym discovery. For each dark jargon, the interpreter takes the following steps to uncover its hidden meaning. First, we look at the roots of the hypernym candidates forest. If none of them is a valid hypernym of the jargon, as determined by the classier, we label the jargon "unknown". Otherwise, we choose the most probable root (again based upon the output of the classifier) and continue to analyze its children. If none of them is found to be a hypernym for the jargon, their parent is returned. Otherwise, the most probable child is picked and the same procedure is followed recursively on its subtree.

5 Evaluation

5.1 Experiment Setting

In our study, we ran our implementation of Cantreader on 375,993 communication traces collected from four underground forums, using an R730xd server with 40 Intel Xeon E5-2650 v3 2.3GHz, 25M Cache cores and 16 of 16GB memories. Here we describe the datasets used in the study and the parameter settings of our system.

Datasets. We used four datasets in our study: dark corpora, benign corpora, hypernymy dataset, and groundtruth dataset with known jargons.

• *Dark corpora*. Dark corpora consist of communication traces from the four underground forums. In our research, we parsed the underground forum snapshots collected by the darknet marketplace archives programs and other research projects [3, 31], to get four dark corpora: the Silk Road corpus [17] consists of 195,403 traces (i.e., threads of posts) from the underground market-

place Silk Road mainly discussing illicit products (such as drugs and weapons) trading; the Darkode corpus [4] includes 7,417 traces from a hacking technique forum about cybercriminal wares; the Hack forums corpus [9] has 52,670 traces from a blackhat technique forum; and the Nulled corpus [12] contains 121,499 traces from a forum talking about data stealing tools and services. Table 3 summarizes the dark corpora we used.

• *Benign corpora*. As mentioned earlier, Cantreader uses two different benign corpora: a legitimate reference corpus, and a reputable interpretative corpus. In our implementation, traces of Reddit are used as the legitimate reference corpus. Reddit is the most popular forum in the U.S., receiving around 470,000 posts per day during the past ten years [15]. It includes rich informal language elements in English such as forum slangs, common acronyms, and abbreviations (e.g. "hlp" for "help" and "IMO" for "in my opinion"), which are also commonly used in the underground forums, and therefore it serves as a good reference corpus. To build this corpus, we ran a crawler that scraped 1.2 million traces from 1,697 top subreddits in terms of the number of subscribers [27]. Also, Wikipedia [10] is used as the reputable interpretative corpus. This is because it is large, comprehensive, formally written, and reputable. Table 3 presents the benign corpora used in our research.

• *Hypernymy dataset*. The interpreter component of Cantreader needs a labeled hypernymy dataset to train its classifier. In our implementation, we reuse the hypernymy dataset that Shwartz et. al generate in the previous research [7, 42]. The dataset is constructed by extracting the entity pairs with "is-a" relation from 4 lexical/ontology databases: WordNet [23], DBPedia [5], Wikidata [20] and Yago [45]. It includes 14,135 positive hypernym relations and 84,243 negative ones.

• *Groundtruth dataset*. The groundtruth dataset, with 774 known dark jargons and their corresponding hypernyms, is used in the evaluation of our system. The dataset was collected from two sources: DEA drug code words list [6] and the cybercrime marketplace product list [31]. The DEA drug code words list is the drug jargon list released by Department of Defense Drug Enforcement Administration (DEA), which includes 1,734 drug code words. The cybercrime marketplace product list is a dataset published by academic researchers, which includes 1,292 illegitimate products manually annotated from Nulled, Hack Forums, and Darkode. Note that not all the terms appear on the two lists are actually used as dark jargons in our dark corpora because DEA's drug list includes many out-of-date and uncommon slang names for drugs, and the cybercrime marketplace product list, on the other hand, focuses mostly on illegitimate products, which are not always referred to in dark jargons. Thus we carefully analyzed these terms with the traces containing them and generated a set of 774 groundtruth dark jargons and their corresponding hypernyms of high confidence.

Table 4: Thresholds for different models

SCM	th_{c1}	th_{c2}	th_{c3}	th_{c4}
Silk Road vs. Reddit	0.094	0.161	0.184	0.214
Cybercrime Corpora vs. Reddit	0.086	0.142	0.182	0.209
Reddit vs. Wiki	-0.039	0.0865	0.127	0.154

Parameter settings. In the experiments, the parameters of our prototype system are set as follow:

• *Neural network settings*. We used similar SCM training parameters as shown in Table 1, except that we set `iterations` = 100. We also used `num_wc` = 20 with a window size = 5) to replace the `min_count` parameter due to larger corpora.

• *Thresholds*. Table 4 lists the thresholds we used in our experiments.

• *Projection learning classifier.* We implemented the projection learning with *scikit-learn's* [16] *RandomForestClassifier*. The classifier was trained with the following settings: `n_estimators` = 200, `max_features` = auto, `min_samples_split` = 2, and `class_weight` = balanced.

5.2 Evaluation Results

Accuracy and coverage. In our study, we ran our system over the dark corpora and benign corpus across 1,497,735 traces and 117M words. Altogether, Cantreader automatically identified 3,462 dark jargons and their hypernyms. To understand the accuracy and coverage of the results, we first used the groundtruth dataset to validate our results. Among the 774 jargon words in the groundtruth set, 598 were successfully detected by Cantreader, which gives a recall of 77.2%. We carefully checked the false negatives (i.e., jargons in the groundtruth set but being considered non-jargons by Cantreader), and found that some of false negatives do not show any semantic inconsistency in our corpora. This might be due to the limitation of our corpora, or those terms were not used as jargons during our monitoring timespan. For example, we carefully investigated all the communication traces involving the jargon "car" labeled by DEA. No indicator shows that it is used as "cocaine". In fact, DEA drug code words lists also announce the possible and invertible dataset error due to the dynamics of drug scenes [6]. For the rest 2,864 dark jargons detected by Cantreader, we randomly picked 200 samples for manual validation, where 182 terms were confirmed to be true dark jargons. It concludes that Cantreader achieves a *precision* of 91%.

Performance. To understand the performance of

Table 5: Running time at different stages

stage	running time	traces per second
the discoverer	17.09 hr	2.94
the interpreter	203.33 s	889.68
overall	17.15 hr	2.93

Cantreader, we measured the time it took to process 180,899 communication traces (containing 100M total words, where 75,419 unique words are in the vocabulary) in the dark corpora and the breakdowns of the overhead at each analysis stage, the discoverer and the interpreter. In the experiment, our prototype was running on our R730xd server, using 30 threads. It took around 17 hours to inspect 180,899 traces, as illustrated in Table 5. The results provide strong evidence that Cantreader can be easily scaled to a desirable level to process a massive amount of underground forums every day.

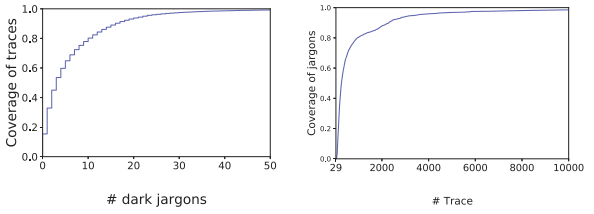
6 Measurement

6.1 Landscape

Scope and magnitude. In total, Cantreader identified 3,462 dark jargons and their corresponding hypernyms from 1,497,735 underground communication traces. Our study shows that criminals indeed widely use dark jargons for underground communication. 376,989 (25%) of the traces include at least one dark jargon. Figure 6a illustrates the cumulative distributions for the number of dark jargons per communication trace. We observe that 80% of the traces using the number of dark jargons less than ten. Later, we study the trace volume of dark jargons. Figure 6b shows the cumulative distributions for the number of communication traces per dark jargon. We observe 80% of dark jargons used by less than 956 traces. It also indicates the effectiveness of our model to capture dark jargons leveraging limited communication traces.

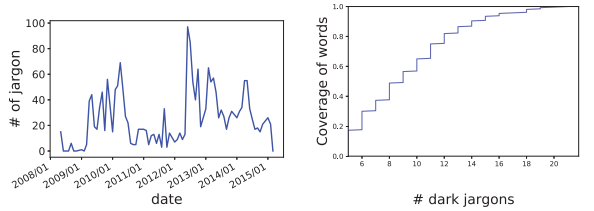
Table 6 presents the 5 categories of dark jargons found by Cantreader in terms of their popularity. We observe that a large portion of dark jargons is drug, which is related to 736 innocent-looking terms. Among them, 692 drug jargons are not included in the drug jargon lists reported by DEA (see Section 5), but prevalent in underground forums such as "cinderella", "pea" and "mango". For example, "mango", the jargon for "marijuana", was found in 540 criminal communication traces about drug trading.

We looked into the distribution of dark jargon across different underground forums. We found that Silk Road has most jargons (2,570). When it comes to the diversity, the communication traces in all four forums include the aforementioned five popular types of dark jargons. This indicates that various kinds of malicious activities



(a) Cumulative distribution of the number of jargons per trace

(b) Cumulative distribution of the number of traces per jargon



(c) Newly-appeared dark jargons in Hack Forums

(d) Cumulative distribution of the number of similar jargons per black word

Figure 6: Characteristic and implication of dark jargons.

Table 6: Dark jargons in categories

category	# hypernyms	# jargons	# traces	examples of jargons
drug	304	736	830,270	blueberry, popcorn, mango
person	1,517	591	460,261	stormtrooper, zulu
software	300	650	512,379	athena, rat, zeus
porn	1	33	2,926	cheesepizza, hardcandy
weapon	672	80	12,055	biscuit, nine, Smith
others	-	1,372	479,789	liberty, ats, omni

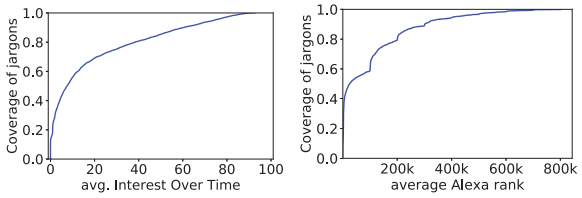
discussed on the underground forums tend to use dark jargons to protect their communication.

Innocent-looking dark jargon. To understand how the criminals choose dark jargons, we study their explicit/innocent meanings. Specifically, we regard the terms' interpretations in WordNet [23] as their innocent meaning, and then seek their nearest common ancestors in the hypernym tree to determine their categories. Table 7 shows the innocent meanings of top 8 jargon categories (in terms of instance number). Most of the dark jargons are deliberate typos and abbreviations (28.24%), followed by person names (4.10%) and locations (3.38%). Interestingly, we found that drug dealers tend to use drug flavors as jargons, e.g., "pineapple", "blueberry", "orange" and "lemon". Meanwhile, hackers prefer mythological figures like "zeus", "loki" and "kraken".

Figure 7a shows the Google search interests of dark jargons when they are used as search terms. Google search interest is recorded by Google Trend [8], which measures the number of searches for each keyword during a time period. The higher search interests means

Table 7: Top 8 categories of innocent-meanings of dark jargons

Type	# jargons	examples of jargons
acronym and abbreviation	910	cp, b1g, delt
name	132	bob, kyle, freeman
location	109	madison, southwest, florence
animal	102	rat, hound, pony
fictional character	56	zeus, loki, pluto
plant	39	lavender, oak
food	31	blueberry, popcorn, cheesecake
vehicle	30	wagon, dandy



(a) CDF of Google Trend's interest score

(b) CDF of search result rankings

Figure 7: Innocent-looking dark jargons.

the higher competitiveness of a search term, indicating that it becomes more difficult for less relevant and less reputable websites to get to the top of the search results under the term through SEO. Due to the generality of the dark jargons' meanings, most of the them (60%) have high search interests. In fact, almost all the top 10 Google search results for each of the dark jargons we found are reputable websites. Figure 7b shows the cumulative distribution of the average websites ranking in search results per dark jargon. We observe that websites in 60% of the dark jargons' search results have the average website rankings below 100k (highly reputable sites). None of them are labeled as malicious websites by Google Safe browsing. This indicates that unlike the black keywords reported in the prior research [46], these dark jargons not only look innocent but are indeed less likely related to compromised or attack sites, thereby providing good covers for underground communication.

Ever-changing dark jargon. Figure 6c shows the evolution of the number of newly-appeared dark jargons on Hack Forums. On average, around 25 dark jargons emerge each month. The trend line in the figure demonstrates two increase tides in 2010 and 2013. Meanwhile, we found that some dark jargons have continued to show up in the communication traces for a long time. For example, "ccs" (credit cards) has been observed from 03/2008 to 05/2016 and is still being used on the underground market.

6.2 Implications of Dark Jargon

Criminal trace identification in benign corpora. The availability of dark jargons enables us to investigate the criminal communication traces on public forums. Specifically, for each communication trace in Reddit, we evaluated whether it includes dark jargons and their co-occurrence terms. The co-occurrence terms are those commonly used in criminal activities on underground forums. For example, "escrow" is a highly-frequent co-occurrence term in the drug advertisement of "blueberry" (marijuana). In this way, we discovered 675 communication traces in Reddit related to criminal activities. Interestingly, among them, 48.3% of the traces with dark jargons do not include their corresponding hypernyms. It means that the criminals intend to use dark jargons to cover its explicit meaning.

To investigate criminal activities of the criminal communication traces using dark jargons, we extracted the keywords using RAKE from the criminal traces and clustered the traces based on those keywords using the classic k-Nearest-Neighbor (k-NN) algorithm [29]. Then, we inspected each cluster manually, and found that most of the communication traces are related to illicit drug trading and drug vendor review. Also interesting, we discovered that drug vendors aggressively post illicit drug trading ads on Reddit: 33 traces about drug trading come from the same vendor *humboldtgrows*. Also, even though Reddit prohibits the posts related to criminal activities [14], we found that the communication traces with dark jargons enjoyed a long lifetime. 73 criminal traces have been there more than one year.

Black words. Cantreader utilizes the semantic inconsistency of dark jargons in the dark corpora and legitimate corpora for identification. However, another type of criminal related terms (called *black words*) are only used by criminals and barely seen on legitimate forums, which cannot be recognized by Cantreader directly. However, we found that such dedicated black words can actually be identified and understood by leveraging the dark jargons we discovered. Specifically, for each word that appears frequently in the dark corpus but has been excluded during the vocabulary building (e.g., due to its absence in C_{legit}, see Section 4.2), we look for its top 40 similar words in the dark corpus (in terms of the cosine distance between word vectors), and examine their overlap with a list of dark jargons we discovered. This jargon list consists of 200 most frequent dark jargons we manually verified. We consider the word to be a black word when the overlap is no less than 5. For such a word, we further used the most common hypernym of the overlapping jargons to interpret it. For example, we found that "chocolope", a kind of marijuana, which does not appear in C_{legit}, frequently co-occurs with multiple drug jargons

Table 8: Top 3 hypernyms with most black words

Hypernym	# black words	percentage
sedative	69	14.4
narcotics	63	13.2
stimulant	46	9.6

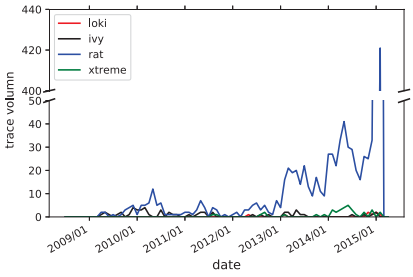


Figure 8: Trace volume of four jargons across month

such as "blueberry", "diesel" and "kush" in C_{dark}. In this way, we discovered 522 black words related to 14 hypernyms. We manually examined and confirmed that 478 were indeed black words, which gives an accuracy of 91.57%.

Figure 6d shows the cumulative distribution of the number of similar jargons per black word. We found that 50% of the black words are similar to more than 10 dark jargons on the list. Table 8 presents the top 5 hypernyms of black words with most instances. Here, "sedatives" have most black words (14.4%), followed by "narcotics" (13.2%) and "stimulants" (9.6%). Also interestingly, criminals utilize obfuscated terms as black words, e.g., "li0n" (crypter) and "Illusi0n" (trojan).

6.3 Case Study

Our research discovers many jargons related to malware or cyber attack services. We found that identifying such jargons helps cyber threat intelligence gathering from the underground forum to better understand various threats. For example, "rat" (remote access trojan) is mentioned in 9,445 unique criminal communication traces, in the contexts of trojan development, new trojan promotion and exploit package purchase/sell.

Figure 8 illustrates the trace volume of the dark jargons "rat", "loki", "xtreme" and "ivy" per month from 05/2008 to 03/2015 in Hack Forum, where "loki", "xtreme" and "ivy" are different kinds of "rat". From the figure, we can see the prevalence of "rat" (including all these jargons) discussion across years, in terms of the number of traces. Overall, 350 traces are related to "ivy", 261 to "xtreme" and 46 to "loki". In fact, compared to "loki" and "xtreme", "ivy" is a more popular "rat" since its release, due to its wide availability and easy-to-use features [13]. We also find that 10% of criminal communication involving "ivy" talks about free download addresses. The traces containing "xtreme" are most for seeking the source code of "xtreme" and its variants. We notice a spree of the trace volume of "rat" from 02/2009 to 10/2011 due to the popularity of multiple kinds of "rat" like "loki", "xtreme" and "ivy". In 02/2015, we observe a small peak of "rat". This is because that Darkcomet 5.3 [2], a kind of "rat", is released and several configuration issues discussions correspond to that.

7 Discussion

Semantic comparison model. As mentioned earlier, we propose a semantic comparison model to address the challenge in comparing the semantics of a word from different corpora. Our current application domain of the model is jargon discovery. We believe that this semantic comparison model could also be used in any polysemy identification scenario if proper corpora exist. We conducted open domain experiments as reported in Section 3, which indicates the effectiveness of the proposed model on open domain corpora.

Also, even only accepting two corpora in jargon discovery, the semantic comparison model can accept n corpora for comparison by setting the input layer to n times of the word size, where the word size is the intersection words of all n corpora. Such scalability offers the effectiveness to processing and comparing multiple corpora at the same time. In fact, we can further optimize the performance of jargon discovery: consider the example mentioned in Section 3; we can modify the semantic comparison model to accept three corpora $legit$, $dark_1$ and $dark_2$ where $dark_1$ and $dark_2$ are related to the similar criminal activity such as drug trading. Then, the model calculates $Sim_{dark_1,legit}$ and $Sim_{dark_1,dark_2}$ and utilizes $Sim_{dark_1,dark_2}$ to further validate the correctness of dark jargon.

Limitation. The main idea of Cantreader is based on the semantic inconsistency of an innocent-looking term in underground communication and in the legitimate corpus. The performance of Cantreader is corpus related. The adversary may play evasion tricks by adding more legitimate terms to their underground communication traces to affect the semantic comparison results. However, even in the presence of relevant content, the dark jargon could still be identified when we select the underground corpora carefully to limit the impact of corpora pollution: such as only selecting a limited number of communication traces from a specific user or including more semantic relevant corpora from different sources. Moreover, our semantic comparison model only conducts word-level semantic inconsistency check and does not support phrase-level jargon detection. A follow-up

step is to optimize the model to identify jargon phrases. A possible solution is to find the possible phrases in underground corpus based on n-gram frequency and concatenate those words into a single term as the input of the semantic comparison model.

8 Related Work

Recently, researchers leverage natural language processing for security and privacy research. Examples include analyzing web privacy policies [49], generating app privacy policies [47], analyzing descriptions to infer required app permissions [40, 39], detecting compromised web sites [34], identifying sensitive user input from apps [33, 38], and collecting threat intelligence [35]. Our work proposes a novel NLP analysis model and identifies a novel application of NLP security, i.e., automatically identifying and understanding dark jargons in underground communication traces.

One recent work that is closest to our study introduces a technique to detect search engine keywords referring to illicit products or services [46]. This work utilizes several search engine result features (such as the number of compromised websites in the search results) to determine whether a search keyword is related to the underground economy. This approach, however, is not suitable for dark jargon detection, because dark jargons are mainly short and innocent-looking terms, which have high search engine competition, i.e., search engine results of dark jargons are mainly highly-reputable websites. Hence, the search engine result features cannot capture dark jargons' illicit semantics but only their innocent semantics. Another relevant work [31] identifies illicit product names in the underground forums. The authors presentes a data annotation methods and utilizes the labeled data to train a supervised learning-based classifier. This work relies on a large amount of human effort for the data annotation and is designed not for dark jargon identification but underground economy product. Also, neither of the previous two works is able to reveal the hidden meaning of the detected dark words automatically. Finally, [48] proposes to use word embedding to analyze the semantics of jargons in Chinese underground market. But their endeavors stopped at a rather initial stage, finding semantically similar words of previous-detected jargons using cosine similarity of embedded vectors. Further manual inspection of those similar words is still required to understand the meaning of dark jargons. Moreover, the author fails to address the problem of how to identify dark jargon from the underground market.

9 Conclusion

In this paper, we present Cantreader, a novel technique for automatically identifying and understanding dark jargons from underground forums. Cantreader is designed to specialize the neural language model for semantic comparison. Our approach can efficiently capture the semantic inconsistency of a term appearing in different corpora, and then further understand such term by identifying its hypernym. Our evaluation of over one million underground communication traces further reveals the prevalence and characteristics of dark jargons, which highlights the significance of this first step toward effective and automatic semantic analysis of criminal communication traces.

10 Acknowledge

We are grateful to Jiang Guo and Dmitry Evtyushkin for their valuable feedback and the anonymous reviewers for their insightful comments. This work is supported in part by the NSF 1408874, 1527141, 1618493 and ARO W911NF1610127.

References

[1] About the Test Data. http://mattmahoney.net/dc/textdata.html.

[2] Darkcomet. https://en.wikipedia.org/wiki/DarkComet.

[3] DARKNET MARKET ARCHIVES (2013-2015). https://www.gwern.net/DNM-archives.

[4] Darkode forum. https://en.wikipedia.org/wiki/Darkode.

[5] DBPedia. wiki.dbpedia.org.

[6] DEA drug code words. https://ndews.umd.edu/sites/ndews.umd.edu/files/dea-drug-slang-code-words-may2017.pdf.

[7] GitHub - HypeNET: Integrated Path-based and Distributional Method for Hypernymy Detection. https://github.com/vered1986/HypeNET.

[8] Google Trend. https://trends.google.com/trends/.

[9] Hack Forums. https://hackforums.net/ .

[10] Index of /enwiki/. https://dumps.wikimedia.org/enwiki/.

[11] NLTK. http://www.nltk.org.

[12] Nulled forum. https://www.nulled.to/.

[13] Poison Ivy. https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-poison-ivy.pdf.

[14] Reddit Content Policy. https://www.reddit.com/help/contentpolicy/.

[15] Reddit statistics. https://redditblog.com/2015/12/31/reddit-in-2015/.

[16] scikit-learn. http://scikit-learn.org/stable/.

[17] Silk Road(marketplace). https://en.wikipedia.org/wiki/Silk_Road_(marketplace).

[18] SPARQL. https://en.wikipedia.org/wiki/SPARQL.

[19] The DEAs list of drug slang is hilarious and bizarre. https://news.vice.com/en_us/article/8xmbpb/the-deas-list-of-drug-slang-is-hilarious-and-bizarre.

[20] Welcome to Wikidata. https://www.wikidata.org/wiki/Wikidata:Main_Page.

[21] Wikidata:SPARQL query service/Query Helper. https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/Query_Helper.

[22] word2vec. https://code.google.com/archive/p/word2vec/.

[23] WordNet. https://wordnet.princeton.edu.

[24] Cybercrime economy: An analysis of cybercriminal communication strategies. https://www.flashpoint-intel.com/blog/cybercrime/cybercriminal-communication-strategies/, 2015.

[25] Underground black market: Thriving trade in stolen data, malware, and attack services. https://www.symantec.com/connect/blogs/underground-black-market-thriving-trade-stolen-data-malware-and-attack-services, 2015.

[26] Cantreader. https://sites.google.com/view/cantreader, 2018.

[27] Reddit metrics: Top subreddits. http://redditmetrics.com/top, 2018.

[28] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[29] G. Casella and R. L. Berger. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

[30] N. Christin. Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22nd international conference on World Wide Web*, pages 213–224. ACM, 2013.

[31] G. Durrett, J. K. Kummerfeld, T. Berg-Kirkpatrick, R. S. Portnoff, S. Afroz, D. McCoy, K. Levchenko, and V. Paxson. Identifying products in online cybercrime marketplaces: A dataset for fine-grained domain adaptation. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[32] R. Fu, J. Guo, B. Qin, W. Che, H. Wang, and T. Liu. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1199–1209, 2014.

[33] J. Huang, Z. Li, X. Xiao, Z. Wu, K. Lu, X. Zhang, and G. Jiang. Supor: Precise and scalable sensitive user input detection for android apps. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 977–992, 2015.

[34] X. Liao, K. Yuan, X. Wang, et al. Seeking nonsense, looking for trouble: Efficient promotional-infection detection through semantic inconsistency search. In *Proceedings of S&P'16*.

[35] X. Liao, K. Yuan, X. Wang, Z. Li, L. Xing, and R. Beyah. Acing the ioc game: Toward automatic discovery and analysis of open-source cyber threat intelligence. In *Proceedings of CCS'16*.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[38] Y. Nan, M. Yang, Z. Yang, S. Zhou, G. Gu, and X. Wang. Uipicker: User-input privacy identification in mobile applications. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 993–1008, Washington, D.C., Aug. 2015. USENIX Association.

[39] R. Pandita, X. Xiao, W. Yang, W. Enck, and T. Xie. Whyper: Towards automating risk assessment of mobile applications. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 527–542, 2013.

[40] Z. Qu, V. Rastogi, X. Zhang, Y. Chen, T. Zhu, and Z. Chen. Autocog: Measuring the description-to-permission fidelity in android applications. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1354–1365. ACM, 2014.

[41] L. Rimell. Distributional lexical entailment by topic coherence. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 511–519, 2014.

[42] V. Shwartz, Y. Goldberg, and I. Dagan. Improving hypernymy detection with an integrated path-based and distributional method. *arXiv preprint arXiv:1603.06076*, 2016.

[43] R. Snow, D. Jurafsky, and A. Y. Ng. Learning syntactic patterns for automatic hypernym discovery. In *Advances in neural information processing systems*, pages 1297–1304, 2005.

[44] R. Snow, D. Jurafsky, and A. Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 801–808. Association for Computational Linguistics, 2006.

[45] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203–217, 2008.

[46] H. Yang, X. Ma, K. Du, Z. Li, H. Duan, X. Su, G. Liu, Z. Geng, and J. Wu. How to learn klingon without a dictionary: Detection and measurement of black keywords used by the underground economy. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 751–769. IEEE, 2017.

[47] L. Yu, T. Zhang, X. Luo, and L. Xue. Autoppg: Towards automatic generation of privacy policy for android applications. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, pages 39–50. ACM, 2015.

[48] K. Zhao, Y. Zhang, C. Xing, W. Li, and H. Chen. Chinese underground market jargon analysis based on unsupervised learning. In *Intelligence and Security Informatics (ISI), 2016 IEEE Conference on*, pages 97–102. IEEE, 2016.

[49] S. Zimmeck and S. M. Bellovin. Privee: An architecture for automatically analyzing web privacy policies. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 1–16, 2014.