Developing Understanding of Hacker Language through the use of Lexical Semantics

Victor Benjamin, Hsinchun Chen
Department of Management Information Systems
The University of Arizona
Tucson, AZ, 85721
vabenji@email.arizona.edu, hchen@eller.arizona.edu

***Abstract*— The need for more research scrutinizing online hacker communities is a common suggestion in recent years. However, researchers and practitioners face many challenges when attempting to do so. In particular, they may encounter hacking-specific terms, concepts, tools, and other items that are unfamiliar and may be challenging to understand. For these reasons, we are motivated to develop an automated method for developing understanding of hacker language. We utilize the latest advancements in recurrent neural network language models (RNNLMs) to develop an unsupervised machine learning technique for learning hacker language. The selected RNNLM produces state-of-the-art word embeddings that are useful for understanding the relations between different hacker terms and concepts. We evaluate our work by testing the RNNLMs ability to learn relevant relations between known hacker terms. Results suggest that the latest work in RNNLMs can aid in modeling hacker language, providing promising direction for future research.**

Keywords - Cybersecurity; Hacker community; Recursive neural network; Language model

I. INTRODUCTION

Cybersecurity is one of the largest issues impacting society. High-profile instances of cybercrime and data theft have become of common occurrence. The whole of society is affected, as we witness attacks targeting individuals, industry, and government. Cybersecurity will remain a problem of great relevance for the foreseeable future. As a result, the need for more research on hackers is a common suggestion in recent years. Specifically, the development of methods to model cyber adversaries is one of the critical but unfulfilled research need outlined in a 2011 report on cybersecurity by the National Science and Technology Council [1]. More research on "black hat hackers", i.e. cybercriminals, would offer new knowledge on securing cyberspace against those with malicious intent, leading to the development of more effective countermeasures against security threats [2].

Many online hacker communities exist that are of interest to cybersecurity researchers. Hackers congregate within online communities to share cybercriminal assets and knowledge [3]. Some communities contain underground economies where participants buy and sell hacking tools and stolen data [4]. However, researchers and practitioners face many challenges when attempting to study hacker community contents. Such communities contain untraditional data much different than more traditional virtual communities. Community participants may discuss hacking terms, concepts, tools, and other hacker-specific items that are unknown to researchers. Foreign language issues may also arise due to hacker communities existing globally, presenting yet another barrier to research.

For these reasons, we are motivated to develop an automated method for understanding hacker language. Specifically, we utilize recurrent neural network language models (RNNLMs) coupled with methodology from lexical semantics to develop an unsupervised machine learning technique for learning hacker language. Unsupervised learning is ideal as feature generation for supervised techniques may be challenging due to the untraditional nature of hacker data, as well as the need for language independent models as hacker communities exist globally. Such a capability would provide great value to the security community by helping help reveal the role or functionality of existing and emerging hacker tools, malware, and other threats.

II. LITERATURE REVIEW

Literature is reviewed from two key areas. Prior work on hacker communities reviewed to provide contextual information on hacker communities. We then review recent works in lexical semantics. In particular, we highlight relevant techniques that can help achieve research goals.

Hacker Community Research

Hackers make extensive use of online communities to support cybercriminal activity. In particular, hackers use such communities to share cybercriminal assets and hacking knowledge with each other [5, 6]. It is not uncommon to witness hacking tools, malware samples, hacking tutorials, and more to be freely shared among community members. An example of such activity can be seen in Figure 1. Additionally, many hackers will share links to other communities, underground economies, and deep web hidden services [7]. Such communities are not limited to a specific geopolitical region, and have been found to exist globally, including areas such as the United States, China, Russia, and the Middle-East [3, 8].

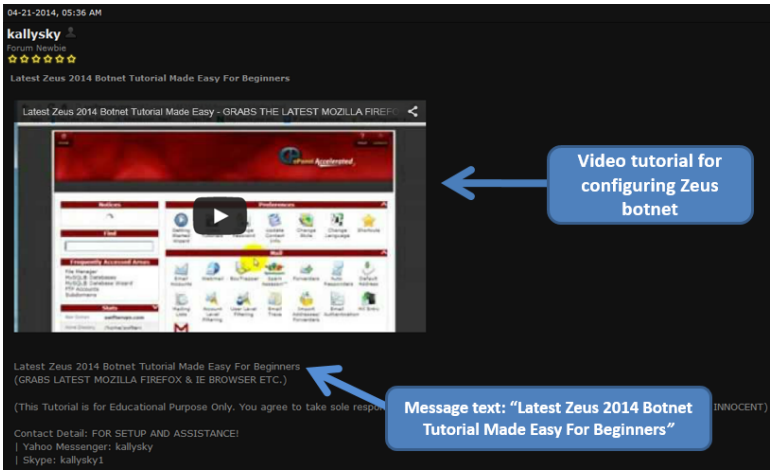

Figure 1 - Example of a posted message on the HackFive.com forum. The message author shares a video tutorial for configuring a popular botnet tool. The message also contains some text describing the video's contents. Such text can be used to build language models that help researchers better understand the role of different hacker terms. For example, here we can observe that "Zeus" refers to a botnet tool.

As a result, recent years have seen security researchers and practitioners develop increased interest in analyzing data from such communities. Past work provides useful methods for identifying and collecting hacker community contents. Additionally, past work provides context and insights for future hacker studies.

Some common methods to identify hacker communities exist throughout literature. Primarily, past studies resort to keyword searches for finding public hacking communities [9]. After an initial set of seed communities are identified, they can be scrutinized for hyperlinks and references to other hacker communities, resulting in a snowball collection procedure [10]. After identification, data can be collected through various means. Forum can be collected with web crawlers; however, anti-crawling measures are sometimes put in place by hacker forums to detect and halt crawling activity [9, 11]. Thus, it may be necessary to use proxy servers and identity obfuscation techniques to avoid detection of crawling activities [6]. For example, adjusting crawling rates and alternating between IP addresses used for crawling hacker contents may help conceal researcher identity and prevent hacker communities from discovering crawling activity.

The majority of previous hacker community research can be categorized within a few major themes. First, much existing work utilizes qualitative analyses to observe and describe hacker community activities [5, 8]. The second branch of work generally involves counting procedures and high-level statistical analyses of underground economy and carding community contents [4, 7, 12]. Lastly, many recent works have focused effort on identifying key participants within hacker communities [6, 13]. These three categories of prior work are useful for describing ongoing activity within hacker communities, as they reveal commonly discussed topics, provide better understanding of hacker social dynamics, and help develop techniques to quickly identify key hacker community participants.

However, one underdeveloped research area is the construction of language models to better interpret hacker contents. Advancements in this area could help boost capabilities for identifying the meaning of hacker-specific terms. Additionally, an understanding of hacker language could help reveal role and functionality of existing and emerging hacker tools, malware, and threats. Lastly, better understanding of hacker language could be used to guide feature generation for future research.

Fortunately, methodology from computational linguistics is useful in text analysis applications. In particular, many prior virtual community studies utilize natural language processing for analyzing web contents. Specifically, methodology from the lexical semantics domain is useful for developing understanding of words and phrases. Such techniques may prove useful for analyzing hacker language.

Lexical Semantics

Lexical semantics is a subfield of linguistics that focuses on: (1) the study of lexical units such as words, affixes, phrases, etc., (2) lexical relations, or how different lexical units relate to each other, and (3) how lexical units map into different concepts. Literature on lexical semantics is far too broad to be discussed in full here, thus we focus on the most recent, relevant stream of work. Specifically, we focus our review on scalable, automated techniques that are suitable for large-scale virtual community research. Additionally, we limit our review to research utilizing unsupervised learning as identifying informative and useful features in untraditional datasets (e.g., hacker communities) presents a difficult challenge. Further, feature-driven techniques are often times language-specific, presenting problems for extending our hacker language modeling to the global scale.

In particular, recurrent neural network language models (RNNLMs) have captured much attention in recent years [14, 15, 16, 17, 18]. They have gained vast popularity due to recent advancements in computing continuous vector representations

of words, which has resulted in high performance and low computational cost relative to other techniques. Additionally, while traditional neural networks that are designed to only feed-forward information through network nodes, recurrent neural networks contain nodes that are interconnected to each other in order to form a directed cycle of information flow. This process creates an internal state within the neural network, causing the model's learning process to become based on previous model states during training. Such behavior is advantageous for learning tasks such as language modeling.

Recent works focus on using RNNLMs to build word embeddings. That is, RNNLMs are used to support unsupervised learning of words by scrutinizing the local context that each word is used within. At a conceptual level, word embeddings simply amount to vectors that contain values representing the local contexts a given word is found within. These vectors, or embeddings, can be used for further computational analyses to extract meaning from unstructured text.

Word embeddings have been researched heavily in recent literature [14, 16]. One major research applications involving word embeddings is to use them for computing the similarity/distance between any two words that are part of the same vocabulary. A second major application is to use word embeddings for learning analogy tasks such as "hat is to head as shoe is to ___ (foot)." Word embeddings are particularly useful for identifying word meaning. They can be used to develop conceptual understanding of unfamiliar terms and to measure how conceptually distant two different words are from each other. Such applications may be useful for advancing our understanding of hacker language.

III. Research Gaps and Questions

There is a growing body of work investigating hacker communities, but none appear to focus on advancing our capability to understand hacker language through automated means. An automated method to identify and learn the significance of different hacker terms, tool names, etc. would be of great asset due to its ability to help identify emerging threats and hacker trends. At the same time, there have been many recent advancements in using recurrent neural networks construct language models through the use of word embeddings. Such a technique may be appropriate for our research context. Thus, we are motivated to utilize the latest work on recurrent neural networks in attempt to build new capability for automatically developing understanding of hacker language. We posit the following research questions:

- How can we develop the capability to automatically digest hacker community contents and learn about the hacker language?
- Is there a possibility for an unsupervised, scalable approach to understanding hacker language?

IV. Research Testbed and Design

Our research design (Figure 2) consists of a series of steps involving automated data processing and analysis. First, we identify and collect hacker forums for this study. Next, we process collected data into a form ready for analysis. We then construct our RNNLMs and execute experiments. Finally, experiment results are evaluated and conclusions are drawn based on our findings.

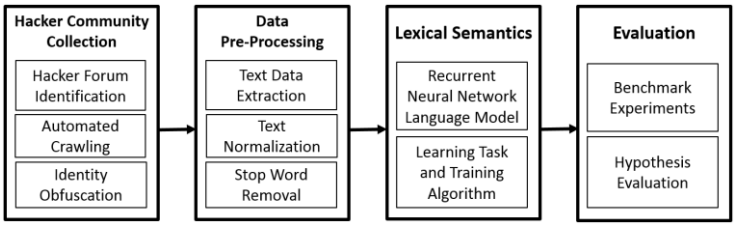

Figure 2 – Research Design

Similar to previous hacker community research, we utilize keyword searches to identify hacker forums to serve as a testbed for this study. For example, keywords included "hacker community" and "blackhat forum." Automated crawlers deployed to collect identified forums. We circumvented potential anti-crawling mechanisms by altering crawling rates to avoid detection. Additionally, we routed our Internet traffic through the *Tor* anonymization network to hide researcher identity and university affiliation. The *Tor* network is a peer-to-peer Internet traffic routing service that effectively anonymizes Internet communications. Packets that enter the *Tor* network are relayed to three or more volunteering peers before reaching their destination; the IP address of the original sender is thus concealed past the first relay, protecting researcher identity from hacker communities.

We identify and collect two hacker forums for this study (Table 1). Forums were chosen based on several factors. First, both forums are English-speaking hacker communities. While the technique we use is language independent, we chose to test our models on English forums as we can more easily interpret results than other languages. Additionally, both forums contain consistent forum activity over time with recent activity from multiple forum participants. Lastly, we observe abundant discussion of hacking concepts and tools that would be interesting to study in this research.

TABLE I. Research Testbed

Forum	Members	Threads	Posts	Time Span
HackFive	947	1,108	5,334	1/24/2013 – 12/30/2014
HackHound	633	507	3,621	12/10/2012 – 12/30/2014

After collection, we extract message text from collected hacker forum web pages. Regular expressions written to extract data embedded within HTML, including thread titles and message bodies. We then normalize extracted messages in preparation for analysis. First, we convert all text to lowercase so that the same word in different cases are not treated as two separate words in our model. Second we strip punctuation from words to again avoid duplication of words. Stop words are also removed from the test bed.

To develop word embeddings for our research, we utilize the RNNLM proposed recently in [16]. This new model has gained traction among many researchers for generating state-of-the-art word embeddings [15, 18]. The selected RNNLM has been benchmarked across multiple studies and shown to be a leading performer for language modeling and information retrieval problems [14, 15, 16].

The selected RNNLM can generate word embeddings via two distinct learning tasks, continuous Bag-of-Words (CBOW) and skip-gram learning [16]. The difference between the CBOW and Skip-gram learning tasks can be described as such: CBOW predicts a word given context, while skip-gram predicts context given a word. For example, assume you have a window of words $\{w_1\ w_2\ w_3\ w_4\ w_5\}$. CBOW predicts w_3 given the surrounding context words w_1,w_2,w_4,w_5. Conversely, skip-gram predicts the context w_1,w_2,w_4,w_5 given w_3. The context window size can be adjusted to tune either learning task.

For the CBOW learning task, it is possible to predict the word *"States"* given only the context {*The*, *United*, *of*, *America*}. The word *"States"* would be w_3 in this example. CBOW effectively results in multiclass classification of test bed vocabulary, with each word acting as a class. It performs particularly well with small context window sizes. In regards to the skip-gram learning task, it would be possible to predict the context {*Recursive, Neural, Language, Model}* when provided only the word "*Network*". In this example, the "*Network"* would also act as w_3, despite the prediction task being different. The choice of which learning task to use is considered application specific [16]. However, CBOW is generally regarded as faster and better for modeling frequent words, while skip-gram is slower to train as it is a more difficult learning task, and is considered better for modeling infrequent words.

However, CBOW and skip-gram only serve as learning tasks, and thus still require approximation algorithms for full implementation. In particular, approximation algorithms are needed to compute the conditional probabilities of all words within a vocabulary, and then to normalize the produced probability distribution. The reason such algorithms are necessary is because the computational complexity of computing the probability distribution for entire vocabulary is $O(n)$, where n is the vocabulary size. This can be extremely problematic for larger datasets, such as virtual community data.

Two popular approximation algorithms used in recent RNNLM work are the hierarchical softmax and negative sampling algorithms [16, 18, 19]. Hierarchical softmax builds a Huffman tree for mapping out the probability distribution for a given vocabulary. This method is considered as better performing for infrequent words. Conversely, negative sampling approximates the probability distribution by sampling data, resulting in better performance for frequent words that are more likely to be sampled. Further, both approximation algorithms are compatible with CBOW and Skip-gram, resulting in interchangeably parts to build the RNNLM as proposed in [16]. However, regardless of model chosen, both reduce time complexity for training the RNNLM; the appropriate choice of which algorithm to use is application-specific, requiring evaluation of model performance.

Unfortunately, evaluation of unsupervised learning is a common challenge in research. Ideally, evaluation can occur by benchmarking the performance of a new technique against proven baseline techniques on a well-known test bed [14]. However, in our context, we want to understand hacker-specific words and concepts using an untraditional dataset that has not been extensively studied. Additionally, we are faced with the challenge of needing evaluation of four separate RNNLM configurations [19]. These configurations include CBOW learning with hierarchical softmax approximation, CBOW learning with negative sampling approximation, skip-gram learning with hierarchical softmax approximation, skip-gram learning with negative sampling approximation.

One recommended evaluation performed in previous research on the selected RNNLM is to compute the similarity between two known words that have been learned by the RNNLM [16]. Since word embeddings are simply word vectors, it is possible to compare the similarity of words by computing the cosine similarity of their embeddings. This procedure effectively reveals how close or distant words are in their meaning. Thus, we can train the RNNLM and test what relationships it has learned between known hacker terms. Results can be evaluated based on our own knowledge of different hacker terms. We design an experiment based on similarity scoring to test RNNLM performance using some known hacker terms:

1. For each of the four model configurations:
2. Using a subset of known hacker terms (e.g., botnet, keylogger), find k most similar words in a separate experiment for each hacker term
3. Score relevancy (i.e., precision) of returned results for each tested term
4. Compare model configurations using precision-at-k

V. HYPOTHESES

Our hypothesis development is primarily guided by prior work on RNNLMs. In particular, recent RNNLMs have been benchmarked with high performance across many different natural language processing tasks, and are considered as state-of–the-art language modeling techniques [15, 16, 19]. This has led RNNLMs to gain traction among many researchers. Thus we posit our first set of hypotheses:

- *H1:* RNNLMs will be useful for developing understanding of hacker language
  - *H1a:* CBOW learning with hierarchical softmax approximation will aid in developing understanding of hacker language
  - *H1b:* CBOW learning with negative sampling approximation will aid in developing understanding of hacker language
  - *H1c:* Skip-gram learning with hierarchical softmax approximation will aid in developing understanding of hacker language

- *H1d:* Skip-gram learning with negative sampling approximation will aid in developing understanding of hacker language

However, the performance of RNNLMs on a given test bed changes based on the model configuration used [17, 20]. As stated earlier, there are four total configurations, with each configuration focused on maximizing performance in different way (e.g., if a model better for frequent or infrequent words). Thus, we posit two hypotheses for this research:

- *H2:* The model configurations will vary in performance
  - *H2a:* CBOW learning with hierarchical softmax approximation will perform the best
  - *H2b:* CBOW learning with negative sampling approximation will perform the best
  - *H2c:* Skip-gram learning with hierarchical softmax approximation will perform the best
  - *H2d:* Skip-gram learning with negative sampling approximation will perform the best

VI. Evaluation and Discussion

We apply each of the model configurations separately against our hacker forum testbed, thus generating four distinct trained RNNLMs. Each RNNLM needs to be evaluated and compared to other competing models. To operationalize evaluation, we perform word similarity experiments as described in our research design. The goal of our experiments is to evaluate performance on real-world application. Specifically, we take a subset of 10 popular hacker terms, and retrieve the 10 most similar words generated per each of the four models. We then calculate a precision-at-10 (P@10) metric for the output of each model by checking if the 10 outputted words possessing high similarity are indeed relevant to the inputted test term.

The 10 words are *botnet*, *RAT*, *card, logger, crypter, rootkit, salt, binder, dork, vulnerability*. *Botnet*, *RAT, and rootkit* refer to hacking tools designed to take control over victim computers. *Card* is related to carding activities and the underground economy. *Logger* refers to keyloggers designed to stealthily capture keystrokes, resulting in theft of passwords and sensitive information. *Crypter, binder,* and *salt* refer to encryption tools. *Dork*, also commonly known as *Google Dorking*, refers to a technique of abusing search engines by searching for HTML snippets belonging to vulnerable web software; in turn, the search engine will return a result list of vulnerable websites. Finally, *vulnerability* is a more general term referring to exploitable security holes.

Each of the 10 test terms served as input for each of the four model configurations. We adjust the context window size for each model based on suggestions from prior work for optimizing performance [16, 19]. Specifically, we use a window size of 5 for CBOW models and a window size of 10 for skip-gram models. In Table 2 and Table 3, we showcase two examples to demonstrate our experiment and evaluation technique. We follow our example with summary statistics over all 10 test terms in Table 4, along with a discussion of our results.

The first input term we test with our models is "botnet." In this case, we assume relevant output would include terms related to botnet tools or other related hacking tools. Output for each model can be viewed in Table 2. Relevant terms are bolded. Similarity refers to the cosine similarity between word embeddings between the test term and results. Along with the abbreviation CBOW for continuous bag-of-words, we abbreviate other terms within the table for formatting purposes: skip-gram is represented as SG, hierarchical softmax as HS, negative sampling as NS, and precision-at-10 as P@10.

TABLE II. Results for Test Term "Botnet"

	CBOW+HS		CBOW+NS		SG+HS		SG+NS	
	Word	*Similarity*	*Word*	*Similarity*	*Word*	*Similarity*	*Word*	*Similarity*
1	Tongue	0.378813	**Citadel**	**0.558807**	Samaritan	0.548317	**Easysocks**	**0.537543**
2	Mint	0.359320	**Zeus**	**0.457285**	Panel	0.543832	Love	0.532371
3	Browser	0.337975	Partners	0.450086	Pony	0.533563	Panel	0.498242
4	Tire	0.335892	**Pandemiya**	**0.430663**	**Phase**	**0.520787**	Satisfy	0.494672
5	Decompile	0.326889	**Mailer**	**0.42337**	Satisfy	0.497737	**Ice**	**0.486135**
6	March	0.323899	Panel	0.407281	Appreciate	0.480637	Record	0.487706
7	Production	0.323290	Linksys	0.374254	**Spyeye**	**0.461338**	Mine	0.484876
8	Opt	0.319973	**Cythosia**	**0.374112**	Fact	0.452641	**Phase**	**0.480187**
9	**Trojanforge**	**0.290963**	**Phase**	**0.369960**	**Cythosia**	**0.441317**	Knock	0.478323
10	**Aircrack**	**0.285836**	**Spyeye**	**0.367168**	**Dendroid**	**0.432320**	**Cynthosia**	**0.475812**
P@10	20%		70%		40%		40%	

For the "botnet" term, the CBOW + NS model produced the most relevant results with 70% P@10. Returned terms such as *Citadel*, *Zeus*, and *Pandemiya* all refer to various botnet tools hackers can retrieve from forums and utilize to infect victim computers and construct their own botnets. The next test term we experimented with was "RAT," which stands for "remote administration tool." RATs are another form of hacking tool that provide cybercriminals with a way to take over full control of infected computers. We expect relevant output to include names of different RATs; results for each model are included in Table III.

TABLE III. Results for Test Term "RAT"

	CBOW+HS		CBOW+NS		SG+HS		SG+NS	
	Word	*Similarity*	*Word*	*Similarity*	*Word*	*Similarity*	*Word*	*Similarity*
1	Moderate	0.404145	**Smallnet**	**0.457760**	**Adwind**	**0.609215**	**Smallnet**	**0.643995**
2	**Adwind**	**0.384969**	**Spygate**	**0.454227**	**Smallwind**	**0.594950**	**Spygate**	**0.623990**
3	Normal	0.346874	**Adwind**	**0.38218**	**Xanity**	**0.591415**	Vast	0.596055
4	**Blacknix**	**0.344942**	Year	0.353762	**Spygate**	**0.583843**	Around	0.573699
5	Saw	0.340433	Love	0.342598	**Blacknix**	**0.566319**	Poor	0.569430
6	Contribute	0.329121	**Njrat**	**0.337081**	Decoder	0.564623	**Njrat**	**0.567891**
7	Clarify	0.324586	**Jrat**	**0.331966**	**Bifrost**	**0.556604**	**Dendroid**	**0.562599**
8	Powerful	0.319936	**Blacknix**	**0.331864**	Vast	0.549580	**Adwind**	**0.551196**
9	Receive	0.315975	**Pandemiya**	**0.328259**	**Bozok**	**0.647259**	Love	0.525843
10	**Jrat**	**0.311750**	**Bifrost**	**0.305260**	**Njrat**	**0.532173**	Cheap	0.522358
P@10	30%		80%		80%		50%	

Here we again observe good performance with the CBOW + NS model and also the SG + HS model. Interestingly, while both models perform well, they capture some different relevant results. For example, *Smallwind* and *Xanity* are two RATs identified by SG + HS, but now CBOW + NS. We run the same form of test for the remaining 8 hacker terms we

identified, and calculate the average P@10 for each model. The results are summarized in Table IV.

TABLE IV. SUMMARY RESULTS

	CBOW + HS	CBOW + NS	SG + HS	SG + HS
Average P@10	20%	70%	66%	56%

Overall, we see evidence that recent work in RNNLMs can be used to develop capability for understanding hacker language, supporting *H1*. In particular, with strong performance from the CBOW + NS and SG + HS models, we see support for *H1b* and *H1c*. Further, the variation in model performance also supports *H2*, as the CBOW + HS and SG + NS model configurations seemed to produce less useful results. The overall top performing model was CBOW + NS with 70% P@10, providing support for *H2b*. However, the SG + HS model closely follows and should also be considered for future work.

We consider some factors that may have skewed our analysis. First, a larger hacker community may alter the performance of our models. It would be useful to collect more data (including from hacker communities of different languages) to further evaluate model performance. Additionally, the 10 selected hacker test terms we utilized may produce biased results; additional testing with more terms would be advantageous. Overall, RNNLMs appear promising for developing understanding of hacker language and advancing our capability to learn the meaning behind different hacker terms, concepts, and their relations to each other.

VII. CONCLUSION AND CONTRIBUTIONS

We use the latest advancements in RNNLM research to develop understanding of hacker language. New RNNLMs are considered state-of-the-art for providing scalable, unsupervised learning of the meaning of words within a corpus through the use of word embeddings. We use this capability to further our understanding of hacker terms, concepts, and relationships between them. Overall, we see promising direction for future research, such as extending our work to include a temporal analysis for identifying the most recent, emerging hacker terms and threats. Such capability would be of great asset for helping security researchers and practitioners learn the latest trends within hacker communities.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation under Grant No. SES-1314631 and also under Grant No. DUE-1303362.

REFERENCES

[1] National Science and Technology Council, "Trustworthy Cyberspace: Strategic Plan for the Federal Cybersecurity Research and Development Program," pp. 1–19, 2011.

[2] M. Siponen, D. Straub, H. R. Rao, and T. S. Raghu, "Moving Toward Black Hat Research in Information Systems Security An Editorial Introduction to the Special Issue," *MIS Q.*, vol. 34, no. 3, pp. 431–433, 2010.

[3] V. Benjamin and H. Chen, "Securing Cyberspace : Identifying Key Actors in Hacker Communities," *IEEE Intelligence and Security Informatics*, pp. 24–29, 2012.

[4] T. J. Holt and M. Kilger, "Know Your Enemy : The Social Dynamics of Hacking," *Honeynet Proj.*, pp. 1–17, 2012.

[5] M. Motoyama, D. McCoy, K. Levchenko, S. Savage, and G. M. Voelker, "An analysis of underground forums," *Proc. 2011 ACM SIGCOMM Conf. Internet Meas. Conf. - IMC '11*, p. 71, 2011.

[6] V. Benjamin and H. Chen, "Time-to-event Modeling for Predicting Hacker IRC Community Participant Trajectory," in *IEEE Intelligence and Security Informatics*, 2014.

[7] J. Martin, "Lost on the Silk Road: Online drug distribution and the 'cryptomarket,'" *Criminol. Crim. Justice*, Oct. 2013.

[8] T. J. Holt, D. Strumsky, O. Smirnova, and M. Kilger, "Examining the Social Networks of Malware Writers and Hackers," *Int. J. Cyber Criminol.*, vol. 6, no. 1, pp. 891–903, 2012.

[9] H. Fallmann, G. Wondracek, and C. Platzer, "Covertly Probing Underground Economy Marketplaces," *Proc. 7th Int. Conf. Detect. intrusions malware, vulnerability Assess.*, pp. 101–110, 2010.

[10] T. J. Holt and E. Lampke, "Exploring stolen data markets online: products and market forces," *Crim. Justice Stud. A Crit. J. Crime, Law, Soc.*, vol. 23, no. 1, pp. 33–50, Mar. 2010.

[11] J. F. Spencer, "Using XML to map relationships in hacker forums," *Proc. 46th Annu. Southeast Reg. Conf. XX*, p. 487, 2008.

[12] M. Yip, N. Shadbolt, and C. Webber, "Why Forums ? An Empirical Analysis into the Facilitating Factors of Carding Forums," *ACM Web Sci.*, vol. May, 2013.

[13] A. Abbasi, W. Li, V. Benjamin, S. Hu, and H. Chen, "Descriptive Analytics : Examining Expert Hackers in Web Forums," in *IEEE Intelligence and Security Informatics*, 2014.

[14] P. Jansen, M. Surdeanu, and P. Clark, "Discourse Complements Lexical Semantics for Non-factoid Answer Reranking," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, pp. 977–986.

[15] O. Levy and Y. Goldberg, "Linguistic Regularities in Sparse and Explicit Word Representations," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning Conference*, 2014, p. 171.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv Prepr. arXiv*, p. 1301.3781, 2013.

[17] A. Mnih and K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," in *Advances in Neural Information Processing Systems*, 2013, pp. 2265–2273.

[18] J. Pennington, R. Socher, and C. D. Manning, "GloVe : Global Vectors for Word Representation," in *Proceedings of the Empiricial Methods in Natural Language Processing*, 2014.

[19] 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*, 2013, pp. 3111–3119.