Cyber Entity Extraction—Evaluating Current Methods, Making the Most of What's Available

Anonymous NAACL submission

Abstract

To address the needs of security analysts, we focus on a crucial element of applied information extraction—accurate identification of basic security entities in relevant text sources. Our survey of previous methods reveals that ground-breaking efforts in this field have not been tested in a realistic setting and that no annotated corpus exists with the characteristics of the targeted security texts. By assembling a representative test corpus, we perform a quantitative evaluation of previous methods in a realistic setting, revealing an overall lack of recall, and giving insight to the models' beneficial and hindering elements. Informed by this evaluation, we propose three novel cyber entity extractors, each of which increases the state of the art in recall, with maximal or near maximal F1 score. Our results establish that the state of the art in cyber entity tagging is characterized by F1 = 0.61.

1 Introduction and Background

Timely and historical security information resides in a variety of online sources, such as security blogs, news documents, vendor bulletins, forum posts, and databases. Moreover, network security analysts depend on these documents for understanding their assets' vulnerabilities, prioritizing patches, tracing clues during forensic efforts, and understanding emerging threats. As evidenced in previous works, off-the-shelf Natural Language Processing (NLP) tools are incapable of isolating security-relevant information, as such tools are configured to identify more general entities and relationships, e.g. people, places, dates (Bridges et al., 2014; More et al., 2012). Hence, the current practice for retrieval of security information from unstructured text documents is still a predominantly manual process, and automated extraction methods for cyber-related information from text are nonexistent but sorely needed.

Recently, research efforts to explore consequential new capabilities made possible by successful cyber information extraction are appearing in the literature (Iannacone et al., 2015; More et al., 2012; Mulwad et al., 2011; Ritter et al., 2015). A compelling example of the potential benefits from accurate information retrieval of cyber security concepts is illustrated by McNeil et al. (2013), who identify a public disclosure of a Microsoft Windows vulnerability and exploit in a series of tweets and blog discussions months before incorporation into standard vulnerability databases or acknowledgment by the vendor. The common thread of these works is the reliance on NLP to identify necessary, security-relevant information from online sources and transform the text into a structured, programmatically accessible, and efficiently human-understandable format.

To address this need, a growing body of research seeks to tailor NLP techniques for the cyber domain (see Table 1) (Bridges et al., 2014; Jones et al., 2015; Joshi et al., 2013; Lal, 2013; McNeil et al., 2013). Accurate identification of basic security entities is a

Table 1: Previous works involving cyber entity extraction techniques

Publication	Type	Algorithm	Training Corpus
(McNeil et al., 2013)	Semi-Supervised	Bootstrapping	Unannotated
(Joshi et al., 2013)	Supervised	Maximum Entropy	Hand Annotated ∼50k Tokens
(Bridges et al., 2014)	Supervised	Maximum Entropy	Auto Labeled ∼750k Tokens
(Jones et al., 2015)	Unsupervised + Distant Supervised	Heuristics	N/A

crucial first step for all such research and is the focus of this work.

1.1 Previous Cyber Entity Extraction Works

Supervised techniques for named entity recognition (NER) are thoroughly developed, and the works of Bridges et al. (2014) and Joshi et al. (2013) attempt to leverage this powerful machinery for cyber entity extraction. [1] The main hurdle of these works is creation of an annotated corpus necessary for training. While Joshi et al. painstakingly hand label a corpus of documents, Bridges et al. craft a set of heuristics that leverage the structured data of the National Vulnerability Database (NVD) [2] to label associated text, and the result is a large automatically-labeled corpus. Both works propose similar supervised techniques trained and tested on their respective corpora, and both exhibit impressive precision and recall results in their testing environments.

In addition to the supervised methods above, Jones et al. (2015) implement a cyber entity tagger as a preprocessing step for a relation extraction framework. Following the automatic labeling techniques established by Bridges et al. (2014) to create an annotated training corpus, Jones et al. query Freebase[3] to create gazetteers of software information, and reconfigure the heuristics of Bridges et al. to label cyber entities in arbitrary documents. While Jones et al. give evaluation results for their relation extraction algorithm's precision, no evaluation of their entity tagger is discussed.

McNeil et al. (2013) implement a semi-supervised bootstrapping algorithm that both learns heuristics for identification of cyber entities and learns instances of the entities by iteratively cycling through a large unannotated corpus. While we do not build on this clever method to automatically generate heuristics, we do incorporate heuristic-based extraction techniques in both our evaluation of previous works and our development of a new model.

1.2 Novel Contributions

The goal of this paper is twofold: (1) to understand the efficacy of cyber entity extraction methods in a realistic scenario (i.e., using documents from a variety of diverse security-relevant sources), and (2) to produce new cyber entity extraction capabilities that leverage all the resources available to provide optimal extraction capabilities (maximal F1 score).

While investigating the previous two supervised cyber extraction works (Bridges et al. (2014) and Joshi et al. (2013)), we observe that the training corpora are predominantly comprised of text from databases and vendor sites; consequently, both have limited scope and use writing styles not representative of the text security analysts target (blogs, news documents, forums, ...). As both models employ features designed to encode the structure of the text, we hypothesize that these features will produce models overfit to nonrepresentative training sets, and yield inadequate labeling when used in the wild. To test this hypothesis and provide a realistic evaluation scenario, we create an evaluation corpus by hand labeling a set of 24 documents gathered from a wide variety of security relevant text sources. Next, implementation of models using the previously proposed features are trained and tested on both held out test sets and the representative corpus (Section 3).

Following the work of Jones et al. (2015), we implement a heuristic tagger based on the automatic labeling techniques from Bridges et al. (2014), and compare its tagging precision and recall against the

[1] We note that Lal's thesis (2013) provides the machine learning details for the Joshi et al. (2013) paper.

[2] The publicly available National Vulnerability Database (NVD), `https://nvd.nist.gov/`, gives additional information fields to entries of the Common Vulnerabilities and Exposure (CVE) database `https://cve.mitre.org/`.

[3] Freebase, `www.freebase.com` is a publicly available database.

supervised models on the representative test corpus. The results are a bit surprising and our comparison illuminates feature types that have proven a hindrance and others a boon for cyber entity extractors.

Informed by this evaluation, we explore novel cyber entity extraction methods that leverage all of the resources available (Section 4). Specifically, we investigate what features of the supervised models to omit, keep, and create in order to enable accurate application to general documents while training on the (publicly available but limited scope) annotated corpus introduced by Bridges et al (2014). Further, we experiment with using heuristic methods in tandem with supervised methods to exploit the high precision exhibited by heuristic methods but boost recall. Our results show the current state of the art in cyber entity extraction is characterized by F1 score of 0.61, and we exhibit four extractors achieving nearly maximal F1 score. The results illustrate how features can be configured to trade precision for recall while maintaining F1.

In summary, this work builds on the cyber entity extracting literature by making the following contributions,

- We provide the first quantitative evaluation and comparison of the cyber entity extraction methods from the literature on a realistic set of documents.
- We identify problematic issues of the previous supervised methods' training sets and features, and illustrate the accurate extraction made possible by relatively simple heuristic methods.
- We experiment with new feature sets and tandem taggers to find the best possible cyber entity extractor.
- We establish that the state of the art is characterized by maximal F1 score, and give models that exhibit precision/recall tradeoffs while maintaining near maximal F1 score.

2 Prerequisites

In order to create a realistic setting for head-to-head comparison of extraction algorithms, we require a common set of entities, a single representative test corpus, and basic understanding of the previous algorithms. Below we give details of these prerequisites.

Table 2: Entity Categories Used for Evaluation

Entity Type	Example(s)
SW_Vendor	Macintosh, Adobe
SW_Product	Safari, Acrobat
SW_Version	7, 11.0.08
CVE_ID	CVE-2014-1127
MS_ID	MS-14-011
SW_Symbol	mAlloc(), reg.exe

2.1 Common Labels

Before evaluation, we require a core set of well-defined entity types that are integral to the security domain. Our categories follow those of Jones et al. (2015) and Bridges et al. (2014), which agreed, and map readily to the cyber security ontology of Iannacone et al. (2015). See Table 2 for the complete list of labels used for evaluation. Both software applications and operating systems references are given the common label "SW_Product". The "SW_Symbol" category refers to elements of code such as filenames, functions, methods, and classes. "MS_ID" is a Microsoft Bulletin identifier, and "CVE_ID" refers to the vulnerability identification used by CVE and NVD. While the categories of Joshi et al. (2013) captured much of the same information, they are organized differently, and the labels are not directly transferable. We note that Jones et al. and Bridges et al. included a category for "Vulnerability Relevant Term", which parallels the union of "Attack", "Means", "Consequences", and "Network Terms" labels of Joshi et al. Ground truth for these categories is ambiguous; for example, deciding if a word or phrase is indeed relevant to understanding vulnerabilities is subjective. To promote fair evaluation, in this section we have omitted categories for which ground truth is ambiguous, although we agree that in practice such categories can extract important details.

2.2 Test Corpus

To facilitate evaluation in a realistic scenario, we sampled documents from prominent and fairly obscure sources of security information including news

Table 3: Test Corpus Information

Source	Type	Docs.	Tokens
Arstechnica	News	3	1,884
Computer World	News	2	1,250
ASL IT Security	Blog	2	39
cons0ul	Blog	2	566
Twitter	Tweet	9	150
Krebs on Security	Edited Blog	2	929
Openwall	Mail List	2	575
Threatpost	News	2	740
Total		**24**	**6,133**

documents, well-edited blogs (e.g., Krebs on Security), unedited blogs, mailing lists, and tweets. Our goal in creation of the corpus was to produce a set of documents that are representative of the text security analysts must manually consult for security information. Although CVE, NVD, and vendor sites, such as Microsoft Bulletin,[4] provide text descriptions that discuss important information, they are accompanied by structured databases. Automated text extraction is not vital for these sources, and we do not include them in our test corpus. Lastly, we ensure no two documents discuss similar topics to prevent any bias. See Table 3 for test corpus details.

2.3 Maximum Entropy Models

Maximum entropy models (MEMs) are the state of the art in labeling tasks, and they excel because they admit a wide variety of features; for example, features can be designed to encode parts of speech, word capitalization, and combinations of other features. As previous supervised cyber entity extraction works used MEMs, a short introduction is provided.

Both Bridges et al. (2014) and Joshi et al. (2013) employ MEMs. To perform the labeling, features are created that can depend on all the words in the sentence and on the predicted labels of the previous words in the sentence.[5] More specifically, denoting a sentence $\bar{w} = (w_1, \ldots, w_n)$ and supposing we have labeled the first $j-1 < n$ words with labels $\bar{t} = (t_1, \ldots, t_{j-1})$, the probability of tag t_j being assigned to word w_j is given by

$$p(t_j|\bar{w}, \bar{t}) \equiv \frac{\exp(f(\bar{w}, \bar{t}, t_j) \cdot v)}{z(\bar{w}, \bar{t})} \qquad (1)$$

where $f = (f_1, \ldots, f_m)$ denotes the feature vector, $v = (v_1, \ldots, v_m)$ are feature weights to be learned from the training data, and $z(w,t) \equiv \sum_{\hat{t}} \exp[f(w,t,\hat{t}) \cdot v]$, i.e., z is the appropriate constant to ensure the sum over Equation 1 over all possible tags is one. An example of a feature (i.e., a component of the feature vector f) is

$$f_1(\bar{w}, \bar{t}, t_j) = \begin{cases} 1 & \text{if } \begin{array}{c} t_{j-1} = \text{SW\_Vendor}, \\ \text{and } w_{j-1} = \text{“the”}. \end{array} \\ 0 & \text{else.} \end{cases}$$

Once features are set, the model parameters v (the feature weights) are fit to a given annotated training corpus. Common fitting algorithms involve optimization of the likelihood estimate, or online methods such as the averaged perceptron (Collins, 2002; Freund and Schapire, 1999). Traditionally (and in the Bridges et al., Joshi et al. works) bigram and trigram features are used to encode the sequential elements of text.

3 Evaluation of Previous Methods

In this section we describe our implementation of the previously proposed entity extraction algorithms and present results evaluating each algorithm on a common set of documents. Our aim is to test the algorithms in a realistic scenario and facilitate direct comparison. Further, we investigate two questions—(1) Are the supervised methods in the literature overfit to a non-representative test corpus? and (2) How effective is simply using well-crafted heuristics and publicly available lists of cyber entities for extraction? En route to these answers we hope to illuminate what features of the text are most indicative of cyber entities in support of the greater goal of finding the best possible extractor with the available resources.

3.1 Previous Supervised Models

As described in Lal's thesis (2013), Joshi et al. employ CoreNLP (see Manning et al. (2014)), an off-the-shelf tool for training an entity recognizer on a given labeled corpus. While convenient, the use of

[4] `technet.microsoft.com/en-us/security/bulletin/`

[5] We note that CoreNLP, the framework used by Joshi et al., supports features that can depend on all labels in the sentence (not just previously labeled words) although no such features were used.

Table 4: Features Types of Maximum Entropy Models

Feature Type	Joshi et al.	Bridges et al.	Bag of Words	All
Bias	✓	✓	✓	✓
Prefix/Suffix	w_j	(None)	w_j	w_j
Unigram	w_{j-1}, w_j, w_{j+1}	$w_{j-2}, w_{j-1}, w_j, w_{j+1}, w_{j+1}, p_{j-1}, p_j, p_{j+1}, t_{j-2}, t_{j-1}$	w_j, h_j	$w_{j-2}, w_{j-1}, w_j, w_{j+1}, w_{j+2}, p_{j-2}, p_{j-1}, p_j, p_{j+1}\ t_{j-2}, t_{j-1}, h_j$
Bigram	(w_{j-1}, w_j), (w_j, w_{j+1}), (t_{j-2}, t_{j-1})	(w_{j-1}, t_{j-1}), (p_{j-1}, w_j)	(None)	(p_{j-1}, w_j), (t_{j-1}, w_j), (t_{j-2}, t_{j-1}), (h_{j-1}, h_{j+1})
Gazetteer	SW_Vendor, SW_Product (*)	SW_Vendor, SW_Product	(N/A)	(N/A)
Regex-1	(None)	$w_{j-2}, w_{j-1}, w_j, w_{j+1}, w_{j+2}$	w_{j-1}, w_j	$w_{j-2}, w_{j-1}, w_j, w_{j+1}, w_{j+2}$
Regex-2	(None)	(None)	w_{j-1}, w_j	$w_{j-2}, w_{j-1}, w_j, w_{j+1}, w_{j+2}$

Note: The notation w_k, p_k, h_k, and t_k denote the k^{th} word, part of speech, heuristic tag, and predicted tag, respectively. Table above lists feature types relative to current index j, e.g., the current word is denoted w_j. "Bias" feature type always fires and biases the model towards the frequency of the labels. "Prefix"/"Suffix" features match the first/last six characters of the word. "Regex-1" refers to a collection of eight regular expressions, namely, first letter capitalization, interior letter capitalization, first character digit, interior character digit, punctuation, underscore, left parenthesis, right parenthesis. "Regex-2" refers to a collection of five regular expressions targeting CVE and MS Bulletin IDs, and version numbers.
(*) Joshi et al. used a single gazetteer corresponding to their "Software" label, which encompassed both vendors and products. To admit the more specific labels, we use the corresponding two gazetteers.

CoreNLP limits the features available to users. We outline their features in Table 4 along with other models. For training data, a corpus of ∼50,000 tokens is painstakingly hand annotated and is now publicly available.[6] Unfortunately, the labels used in this corpus do not map readily to our labels; in particular, software vendors, products, and some versions are lumped into a single category, "Software". While we cannot evaluate their original model on our test corpus, we do examine a MEM equipped with the same features and trained on the other publicly available labeled corpus introduced by Bridges et al., which shares our labels.

The training corpus created by Bridges et al. (2014) is relatively large (∼750,000 tokens) and is comprised of descriptions from NVD, MS-Bulletin, and Metasploit modules. To produce the annotations, Bridges et al. craft a set of heuristics to match entities in the structured fields of the NVD database with their occurrences in the corresponding text. Next, a set of regular expressions for identifying versions and other entities is used to apply additional labels. The precision and recall of this "automatically labeled" corpus are reported greater than 0.99 and 0.75, respectively, which is acquired by hand-checking a random sample. To facilitate flexible feature creation, Bridges et al. implement their own MEM and craft novel features including regular expressions for words in the sentence and parts of speech. Their implementation used the averaged perceptron for training, which has produced competitive results in accuracy and has proven more effective than maximum likelihood techniques (Collins, 2002; Freund and Schapire, 1999). Table 4 enumerates the complete list of features used. Furthermore, Bridges et al. provide the automatically labeled corpus, their heuristics for automatically labeling the corpus, and their MEM implementation in Python.[7]

We have adapted the Python MEM implementation made public by Bridges et al. for our experiments. All MEMs in this paper used the averaged perceptron training algorithm with five itera-

[6] http://ebiquity.umbc.edu/resource/html/id/355

[7] https://github.com/stucco/auto-labeled-corpus

Table 5: Cyber Entity Extractors' Results

	Model		Held Out 20%			Test Corpus		
	Type	**Features**	**P**	**R**	**F1**	**P**	**R**	**F1**
1	MEM	Joshi et al.	0.97	0.93	0.95	0.67	0.35	0.46
2	MEM	Bridges et al.	0.96	0.94	0.95	0.65	0.40	0.49
3	Heuristics	N/A	N/A	N/A	N/A	**0.80**	0.49	**0.61**
4	MEM	Bag of Words	0.93	0.86	0.90	0.63	0.54	0.58
5	Tandem	Bag of Words	0.83	0.87	0.85	0.61	**0.60**	**0.61**
6	Tandem	All	0.86	0.94	0.90	0.65	0.58	**0.61**

Note: Precision, Recall, and F1 score given. All maximum entropy models (MEMs) fit using five iterations of averaged perceptron training on 80% of corpus of Bridges et al (2014), and results on remaining 20% reported. Heuristic tagger is from Jones et al. (2015). Novel models below midline increase recall while nearly fixing F1. Results show the current state of the art is bounded by F1 = 0.61.

tions through this corpus. For each of the feature sets described in Table 4, we train the model on a randomly selected 80% of the corpus. Table 5 gives results when evaluated on the remaining 20% of the training corpus and on the representative test documents. Considering models 1 & 2 in the table, we observe both MEMs with previously proposed features achieve $\sim 95\%$ precision and recall on the held out section of the corpus, yet both struggle on the actual news, blogs, forum documents—exhibiting $\sim 65\%$ precision and less than 40% recall. This shows that the features are adequate for fitting the training data but produce inaccurate results when used in the wild. Clearly, the differences between the training corpus and real documents are causing the problems, but exactly why these models struggle on the real documents is not obvious. To delve into this issue we investigate heuristics for identifying entities and feature sets that depend less on the structure of the sentence.

3.2 Comparison to Heuristic Tagging

As a preprocessing step to their relation extraction algorithm, Jones et al. (2015) reconfigure the automatic labeling developed by Bridges et al. (2014) (which was used only to annotate a training corpus) to label cyber entities in general documents. This entailed querying Frebase's API to create gazetteers of software vendors and then applying the previously used heuristics. We note that the original work of Jones et al. provided no evaluation of this extraction capability. Using both Freebase and the code made available by Bridges et al. we have recreated this entity tagger, and present its tagging abilities in line 3 of Table 5. The heuristics exhibit $\sim 12\%$ increase in both precision and recall over the previous supervised models, which is perhaps surprising considering the lack of sophistication of the algorithm.

We draw a number of conclusions. First, supervised techniques, although impressive in other domains (e.g., part of speech tagging), are unable to produce adequate results on realistic documents when configured with traditional features, e.g., bigrams of words, and tags. Our evidence suggests these features cause overfitting to the writing styles of the available annotated text. While we have not tested the Joshi corpus (as it has different labels), we expect a similar phenomenon as it is approximately 80% text from vendor sites and vulnerability databases. Next, the increase in both precision and recall exhibited by the heuristics gives insights into what features are indicative of cyber entities; in particular, leveraging available resources such as Freebase, where nearly comprehensive lists of entities reside, is a simple and effective approach. Further, regular expressions, e.g., matching version numbering patterns or camel- and snakecase, are features that gain traction in this domain. Looking forward, this evaluation of previous methods begs the question "How do we create the 'best' cyber entity extractor with the available resources—annotated documents of non representative text, well-developed models for fitting labeled data, very precise heuristics, and publicly available databases on the subject?"

4 Exploring New Models

Informed by the evaluation of previous models, we now seek more effective extraction capabilities with the available resources. While the training corpus may not have a representative writing style, it does include a multitude of security-entities. To leverage this corpus, we seek features that allow a MEM to be ported to general documents after training.

First, we reconfigure the MEM's features to encode less of the sequential structure of the text by considering a bag-of-words model, i.e. only including features that depend on the word to be labeled. Examining the regular expressions used by the heuristic tagger, we add regular expressions of the previous and current word, as many heuristics relied on these. Lastly, we include the heuristic tagger's guess of the current tag as a feature, and omit gazetteer features as they are used by the heuristic tagger. The resulting set of features is given explicitly in the "Bag of Words" column of Table 4. Considering the results (Table 5 row 4), the bag-of-word features MEM has only slightly lower precision than previous MEMs, but yields a higher recall than any previous model. When evaluated on a held-out test set, the bag-of-words MEM performs worse overall than the previous MEMs, which is unsurprising as it has fewer features.

Next, we note that the precision of the heuristic labeling is very high (0.804), meaning when the heuristics have identified an entity, they are correct more than 80% of the time. To harness this, we create a tandem labeler—In a first pass words are labeled using the heuristics. Next, a MEM is used to tag only those words that are not labeled by the heuristics. We test the tandem labeler with two feature sets, the bag-of-words model and a set of features that includes every previously used feature. Our impetus for using all the features is that this MEM is only trained on the words of the corpus that were not identified by the heuristics; hence, words labeled by the heuristics will appear before and after words the MEM attempts to label. To take the positive heuristics into account, we include features that look forward and backward in the "All" feature set (last column Table 4). Columns 5 and 6 of Table 5 give the tandem labeling results. Both exhibit a substantial increase in recall over all previous models, while maintaining F1 score.

Looking over all models, we find a maximal F1 score of 0.61, with four models achieving nearby results. This presents users with a choice of models to weight precision/recall between 80/49% to 60/61%. We note that a number of other experiments were conducted, but not presented to respect length requirements and because the results were not as noteworthy. Using sequential bigrams of the heuristics' labels produced good but not superior results. In additional experiments we add a prior distribution by initializing weights to favor features inspired by the heuristics. Results varied little from using no prior (initiating weights to 0), and inspection of the trained model's weights showed that those features were receiving very high weight regardless of the prior. In summary, our experimentation concludes that for balanced precision and recall the tandem models are superior, and if one requires high precision at the expense of recall, the plain heuristics are best.

5 Conclusion

Our evaluation of the previous models shows that they exhibit low recall. As the previous supervised models achieve impressive results on the held-out training documents, their shortcomings on our representative test corpus is certainly caused by the disparity between the available annotated corpora and the target security source documents. Further evidence for our hypothesis that sequential features are causing the overfitting is given by the results of the new MEM with bag-of-words features. Although it is trained on the same corpus, it achieves comparable precision, and far greater recall than the previous models. We conclude that if only supervised techniques are used on the available corpus, features should encode indicative elements of the word to be tagged in lieu of sequential elements of the text.

Our tandem models give even greater increases in recall and both maintain the maximal F1 score of 0.61. This establishes the current state of the art in cyber entity extraction. Furthermore, as four of the models (the heuristic labeler and the three new labelers) examined obtain near maximal F1, our work offers the operator a tradeoff between precision and recall, albeit discrete.

Looking forward, we predict that to obtain very high precision and recall, a substantial training corpus of documents taken from a variety of sources and hand labeled by annotators seasoned in the security domain will be necessary. Such a corpus presupposes a universally accepted set of labels, and work to develop an ontology of the cyber domain is the topic of research, e.g. (Iannacone et al., 2015) and to some degree already in practice in MITRE's STIX[8] standard. On the other hand, the feasibility of this endeavor is suspect. Aside from human resources needed to attempt such an endeavor, technology evolves so quickly that many entity names will be quickly obsolete while others will not appear in slightly aged texts. With this view, it is perhaps unsurprising that simple heuristic methods, which can leverage regularly updated, nearly comprehensive lists of security entities (sources such as Freebase or Wikipedia) are making substantial contributions to the state of the art in this field. We hope our use of novel features and interplay of heuristics with the traditional, sophisticated methods will assist security-relevant information extraction as the field develops.

References

[Bridges et al.2014] Robert A Bridges, Corinne Jones, Michael Iannacone, Kelly Testa, and John R Goodall. 2014. Automatic labeling for entity extraction in cyber security. In *Third Annual ASE Cyber Security Conference*. ASE.

[Collins2002] Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Freund and Schapire1999] Yoav Freund and Robert E Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine learning*, 37(3):277–296.

[Iannacone et al.2015] Michael D. Iannacone, Shawn Bohn, Grant Nakamura, John Gerth, Kelly M. T. Huffer, Robert A. Bridges, Erik Ferragut, and John T. Goodall. 2015. Developing an ontology for cyber security knowledge graphs. In *Proceedings of the CISRC-10, the Tenth Cyber & Information Security Research Conference*. ACM.

[Jones et al.2015] Corinne L. Jones, Robert A. Bridges, Kelly M. T. Huffer, and John Goodall. 2015. Towards a relation extraction framework for cyber-security concepts. In *Proceedings of the CISRC-10, the tenth Cyber & Information Security Research Conference*. ACM.

[Joshi et al.2013] Arnav Joshi, Ravendar Lal, Tim Finin, and Anupam Joshi. 2013. Extracting cybersecurity related linked data from text. In *IEEE Seventh International Conference on Semantic Computing (ICSC)*, pages 252–259. IEEE.

[Lal2013] Ravendar Lal. 2013. Information extraction of security related entities and concepts from unstructured text. Master's thesis, University of Maryland Baltimore County, May.

[Manning et al.2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

[McNeil et al.2013] Nikki McNeil, Robert A Bridges, Michael D Iannacone, Bogdan Czejdo, Nicolas Perez, and John R Goodall. 2013. PACE: Pattern accurate computationally efficient bootstrapping for timely discovery of cyber-security concepts. In *2013 12th International Conference on Machine Learning and Applications (ICMLA)*, volume 2, pages 60–65. IEEE.

[More et al.2012] Sumit More, Mary Matthews, Anupam Joshi, and Tim Finin. 2012. A knowledge-based approach to intrusion detection modeling. In *Security and Privacy Workshops (SPW), 2012 IEEE Symposium on*, pages 75–81. IEEE.

[Mulwad et al.2011] Varish Mulwad, Wenjia Li, Anupam Joshi, Tim Finin, and Krishnamurthy Viswanathan. 2011. Extracting information about security vulnerabilities from web text. In *Web Intelligence and Intelligent Agent Technology Conference (WI-IAT), IEEE/WIC/ACM*, volume 3, pages 257–260. IEEE.

[Ritter et al.2015] Alan Ritter, Evan Wright, William Casey, and Tom Mitchell. 2015. Weakly supervised extraction of computer security events from twitter. In *Proceedings of the 24th International Conference on World Wide Web*, pages 896–905. International World Wide Web Conferences Steering Committee.

[8] Structured Threat Information eXpression (STIX) is a DHS funded, MITRE effort to provide a structured language to the cyber security domain. `http://makingsecuritymeasurable.mitre.org/docs/stix-intro-handout.pdf`