Recognizing and Extracting Cybersecurity Entities from Text

Casey Hanks [1] **Michael Maiden** [1] **Priyanka Ranade** [1] **Tim Finin** [1] **Anupam Joshi** [1]

Abstract

Cyber Threat Intelligence (CTI) is information describing threat vectors, vulnerabilities, and attacks and is often used as training data for AI-based cyber defense systems such as Cybersecurity Knowledge Graphs (CKG). There is a strong need to develop community-accessible datasets to train existing AI-based cybersecurity pipelines to efficiently and accurately extract meaningful insights from CTI. We have created an initial unstructured CTI corpus from a variety of open sources that we are using to train and test cybersecurity entity models using the spaCy framework and exploring self-learning methods to automatically recognize cybersecurity entities. We also describe methods to apply cybersecurity domain entity linking with existing world knowledge from Wikidata. Our future work will survey and test spaCy NLP tools, and create methods for continuous integration of new information extracted from text.

1. Introduction

Cyber Threat Intelligence (CTI) is data that has been analyzed to attempt to uncover the mechanics of and purpose for a cyber-attack. CTI is vital to cybersecurity professionals for staying up to date on information about new attacks, malware, and the actions of various threat actors. Professionals use this information in a variety of use cases, including campaign tracking, threat monitoring, and actor profiling. However, with a vast amount of CTI being released and updated every day, it is increasingly challenging for human analysts to track the data efficiently and effectively. It is important to develop methods for automated cybersecurity knowledge extraction, to consolidate CTI insights and make it easier for experts to access and use.

Named Entity recognition (NER) is a critical component of automated knowledge extraction, allowing natural language models to label instances of real-world entities in text. To accomplish this, Natural Language Processing (NLP) models must be trained on very large corpora of human-annotated text. There are a number of domain-agnostic text corpora available for training models on generic entity types such as *Person, Organization*, and *Date*. However, general domain entity types are not sufficient for more specialized fields like cybersecurity because they are unable to recognize cybersecurity-specific entities such as malware-type, operating system, or attack-type. These are necessary for downstream tasks like malware analysis, attack and vulnerability classification, and building cybersecurity knowledge graphs (Mulwad et al., 2011; Joshi et al., 2013; Gao et al., 2021; Georgescu et al., 2021). Unlike fields like medicine and law, cybersecurity has few comprehensive training datasets that are available *and* continuously updated.

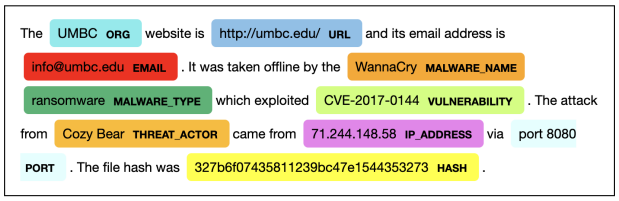

Figure 1. Text showing additional entity types found.

Our goal is to extend existing entity recognition datasets for the cybersecurity domain (Alam et al., 2022; Bridges et al., 2014) to support a general module we call *CyEnts* that can be used in systems that extract cybersecurity information from text. Current public datasets do not recognize entities such as *threat actors, campaigns, or malware*, imperative for training machine learning systems that aim to understand fine-grained threat actor tactics, techniques, and procedures (TTPs). The kinds of entities we need to identify are also more general and diverse than the "named entities" sought by current NLP systems designed to extract information from general domains.

Cyber analysts typically rely on both structured sources like Common Vulnerabilities and Exposures (CVE) records, as well as unstructured, free-text blogs like security vendor blogs to obtain actionable intelligence. Most of the existing entity recognition datasets for the cybersecurity domain target structured data, as the entities are formally defined, providing concrete annotation mappings for entity types. Our research focuses on extending entity recognition capa-

[1]University of Maryland, Baltimore County, Baltimore, MD 21250. Correspondence to: Casey Hanks <chanks1@umbc.edu>, Michael Maiden <mmaiden1@umbc.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

Table 1. New cybersecurity-relevant entity types complement the Ontonotes types supported by spaCy and other NLP systems.

Malware_Name	Campaign
Malware_Type	IP_Address
Software_Name	Protocol
Version_Tag	Threat_Actor
Vulnerability	Operating_System
Attack_Type	Hash
Programming_Language	URL
Email	Path
File_Extension	Function
CVE	Port

bilities for unstructured sources, since these typically provide the most recent information for analysts. Before CTI is concatenated into structured formats like CVE records, the information is first released through online media.

To create a diverse dataset for malware analysis, we have obtained articles, blog posts, and vendor reports from eight different sources, covering high level topics such as adversary motives and origination, to technical analysis of malware behaviors and campaigns. We develop a human annotation study to create high-quality training datasets for machine learning based entity recognition models, using the spaCy framework. Table 1 shows the current list of cybersecurity entity types we support in addition to spaCy's OntoNotes types (Pradhan et al., 2007). We take a *data-centric* approach when creating the training dataset, by using intensive evaluation criteria for entity recognition annotations.

2. Related Work

2.1. AI-Based Cybersecurity

The variety and growth in cybersecurity exploits, vulnerabilities, and threat actors has encouraged the integration of AI-based cyber defense systems for safeguarding critical systems (Wirkuttis & Klein, 2017). The goal of these systems is to provide actionable and relevant insights to analysts that require the information for immediate operation. This is becoming especially critical, as the diversity and amount of Cyber Threat Intelligence (CTI) is rapidly expanding (Tounsi & Rais, 2018). CTI is often *open sourced* and can include data sources such as application logs, malware binaries, network traffic data, and unstructured and semi-structured text. This data is collectively shared across multiple vendors, researchers, and cybersecurity professionals for enhanced situational awareness and intrusion detection and prevention. Examples of threat sharing platforms include MISP (Wagner et al., 2016), STIX (Oasis, 2021), and TAXII (Connolly et al., 2014).

In this paper, we focus on leveraging textual cybersecurity data, which can be commonly found across security blogs, the dark web, and social media. There have been several systems that have been developed that transform free-text cybersecurity into more structured formats for AI-based cyber defense system usage (Samtani et al., 2020b; Mittal et al., 2016; Arnold et al., 2019; Ranade et al., 2021). In particular, several Cybersecurity Knowledge Graphs (CKGs) have been developed to represent disparate CTI data in machine-readable formats, and are used as training data for machine learning systems (Mittal et al., 2019; Pingle et al., 2019; Piplai et al., 2020).

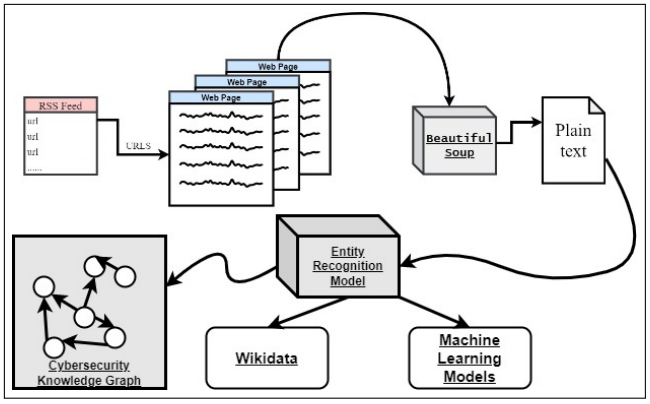

Figure 2. Diagram of the proposed complete system

2.2. Knowledge Representation

Ontologies are one of the primary building blocks of the Semantic Web (Berners-Lee et al., 2001). When populating Ontologies with real world data, information can be associated together and reasoned over through the *web of linked data* (Berners-Lee et al., 2001). Examples of popular knowledge graphs include DBPedia and Wikidata (Vrandečić & Krötzsch, 2014). Machine learning and semantic technologies are more recently, jointly used together for tasks such as language modeling (Agarwal et al., 2020), question-answering (Wang et al., 2016), and information retrieval (Wise et al., 2020).

3. System Overview

CyEnts will incorporate a continuous, up to date corpus of ready to train entity recognition datasets, achieved through a combination of web scraping and entity recognition techniques. This data can then be used to form triples to populate a knowledge graph, train other machine learning systems, as well as link to Wikidata knowledge to provide extensibility to other interoperable systems. In this section we discuss more of how our system proposes to accomplish this.

3.1. Web Scraping

We use a combination of Python *Requests* and *BeautifulSoup* libraries to scrape web articles that contain CTI from

cybersecurity vendor reports/blogs. We regularly update this corpus through periodic scraping to incorporate new articles, preventing stale training data. Our growing corpus can be considered a gold standard dataset for the cybersecurity community due to two primary factors: the source reputability and their breadth of CTI examples. The first is the popularity of the sources utilized across the greater cybersecurity community. Vendor reports and blogs produced by organizations such as *McAfee, Mandiant, FireEye*, and *Juniper Networks*, describe up-to-date and pertinent vulnerability, attack, and adversary information to aid cybersecurity professionals in mitigating incoming attacks, quickly addressing open vulnerabilities, and thwarting future threats (Samtani et al., 2020a). Secondly, vendor reports and blogs do not follow a standard format and are therefore inherently diverse in their communication of vulnerabilities, exploits, and threats, providing the system with multiple perspectives and examples.

We collect our data through RSS feeds, which list a number of articles on each site, in order of publication. This text is stored with no newline characters to simplify tasking for the human annotations, described in Section 3.2.1. We use spaCy facilities to split the text into sentences, and then group the sentences by topic to produce "paragraphs" using NLTK's textiling tools (Loper & Bird, 2002). We currently have about 25,000 sentences from over 380 text articles[1]

3.2. Entity Recognition

The first step in creating an entity recognition system is to define the entity types to recognize. We have created a set of entity types, listed in Table 1, after extensive literature review and multiple revisions. We include common entity types such as *Software_Name* and *Version_Tag*, as well as more fine-grained types, like *Threat_Actor* and *Malware_Type*. We believe this set of entity types to be gold standard for malware analysis, due to the inclusion of descriptive types for entities imperative for processing and understanding malicious activity and file behavior. For example, the operating system a malware targets, or the threat actor involved, can provide additional real-world context to a malware analysis task. We have created a spaCy pipeline for recognizing these entity types through use of human annotations of text, and rule based methods, detailed below.

3.2.1. Human Annotations

To create ground-truth annotations, we task a group of six human annotators to recognize entities in the corpus data. The annotators have backgrounds in Computer Science and Cybersecurity. These annotations, along with the rule-based methods described in Section 3.2.2, will later be used to train machine-learning based methods.

[1]Our textual data is available at `https://github.com/UMBC-Onramp/CyEnts-Cyber-Blog-Dataset`

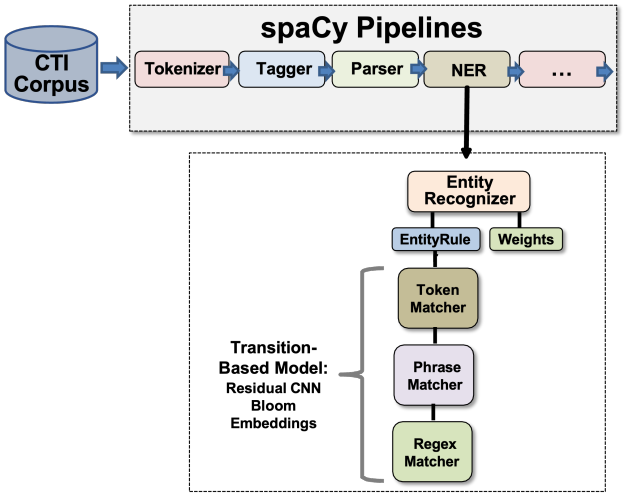

Figure 3. Components of the entity recognition pipeline

To ensure quality annotations, each annotator received training, documentation detailing the definition of each entity type, and multiple in-context examples. We divided the human annotators into three groups. Each annotator within a group was tasked to annotate the same text independently. Once complete, we calculate the Inter Annotator Agreement, which means that only entities that both annotators agreed upon would be accepted into the final dataset. To conduct our annotation task, we use the *Prodigy* annotation tool. This tool allows for seamless integration with the spaCy NLP framework. The resulting annotated compiled dataset is used to train and test an *entity recognizer* model. The architecture of the entity recognizer model we utilized is displayed in Figure 3, and is further explained in the following section. In the first round of annotations each annotator was given the first ten paragraphs of ten randomly selected articles, for a combined total of 1339 annotated sentences. The results from the first round are provided in Section 3.3.

3.2.2. Rule Based Methods

We utilize *spaCy pipelines*, which are several in-built models and libraries utilized for different Natural Language Processing (NLP) tasks. The available pipelines and the specific models we implement are shown in Figure 3. The Entity Recognizer model part of the NER pipeline is utilized for our initial task of extracting and recognizing entities in unstructured CTI. The model employs a transition-based algorithm which utilizes a combination of *Bloom Embeddings* and a Residual Convolutional Neural Network (CNN) (Lample et al., 2016). More specifically, spaCy employs the *Embed, Encode, Attend, Predict* framework (Honnibal, 2016). Each token is first embedded using a *Bloom filter*, which is a hashed embedding dictionary where word hashes,

rather than the actual words, are set as keys.The words are then encoded into a sentence matrix with a Residual CNN. The sentence matrix is further reduced into a single vector to be used in a feed-forward neural network for prediction. This necessarily allows us to lose irrelevant information and attend to the most useful features. We can then use the standard feed-forward neural network for inference, to predict the target representation (entity label).

There are many entity types that can be recognized using rule-based methods. The types fall into two major categories, those that have a finite number of relevant examples and ones with a regular format. For the former category of entities we can use *Gazetteers* (Song et al., 2020) of relevant entities of a particular type. We have developed Gazetteers for the entity types, Operating_System, File_Extension, Attack_Type, Programming_Language, Malware_Type, and Protocol. The lists for the Gazetteers were created by querying Wikidata to find the most prominent instances (direct or inherited) of all of the types, to make sure that we include entities that are relevant. More information on Wikidata entity linking can be found in Section 3.2.3. These lists were then manually reviewed to ensure they were encompassing.

The second category where rule-based methods are helpful are entities which have a strict format, such as IP address, that can be recognized by regular expressions. We use regular expressions with IP_Address, Hash, Port, and CVE vulnerabilities (alongside regular entity recognition for named vulnerabilities). We also use spaCy's built-in rule based recognition for Email addresses and URLs. We included these recognizers in the pipeline used to prime the second round annotations, so that the process was more streamlined.

3.2.3. Wikidata Entity Linking

Where possible, we link entity mentions to items in Wikidata (Vrandečić & Krötzsch, 2014) which has about one billion facts on about 100 million items. It has a Web interface to support exploration and editing by people, a set of APIs to access its information programmatically, and a SPARQL endpoint for querying its RDF knowledge graph. Wikidata's ontology has a very fine-grained type system with more than two million types and over 10,000 properties. The properties of Wikidata items we primarily use are its types, super-types, label, aliases, and description. Many other Wikidata properties are available for future use.

Applying entity linking to a domain involves being able to recognize both the Wikidata items and properties that are very relevant to the domain as well as those that are likely to be irrelevant. We identified custom sets of Wikidata types and properties to support the cybersecurity domain for this project. Given an entity mention, it uses Wikipedia's existing search APIs to retrieve an initial set of candidate entities, typically between 20 and 50. The API searches against the text in labels, aliases, and description as well some property values.

Table 2. Breakdown of number of annotations for each Entity Type

Type	Number	Type	Number
Version_Tag	17	EVENT	1
Malware_Type	33	GPE	36
Threat_Actor	24	LAW	0
Vulnerability	26	MONEY	3
Filename	37	ORG	149
Protocol	29	PRODUCT	1
Port	1	TIME	6
Software_Name	53	DATE	148
Malware_Name	118	FAC	0
Tool	1	LANGUAGE	4
Campaign	14	LOC	0
Operating_System	35	NORP	10
Filepath	7	QUANTITY	0
Process	10	PERSON	16
Attack_Type	2		

The results are then ranked using a variety of matching features, item prominence, and the semantic similarity of the entity's text context compared to its Wikidata description and initial sentence in its DBpedia (Lehmann et al., 2015) abstract. This approach can help CyEnts successfully assign the type Threat_Actor to *Lazarus* in the sentence

"Lazarus was behind the WannaCry attack"

and also link it to the North Korean hacker organization *Lazarus Group* (Q19284445) even though Lazarus matches all or part the names of more than 1,000 Wikidata items.

3.3. Evaluation

The initial Precision, Recall, and F-score for the first round of the NER dataset is provided in Table 3. The breakdown of the Precision, Recall, and F-score for each entity type is listed in Table 4. We have omitted spaCy general types as well as cybersecurity classes with low representation from the table, due to lack of examples present for evaluation. We discuss increasing the representation of sparse labels and tuning the human evaluation study to produce better annotations for training more robust models in Section 3.3.3.

Unlike entity recognition for more structured formats like CVE records, blogs contain a greater degree of variety in sentence structure, information, and style, adding increased difficulty for recognizing relevant entities, in agreeable and universal formats. We are improving our first round of annotations by employing rule-based strategies, in addition to aiding the annotators with further training. We discuss other lessons learned from our first round below.

Table 3. Precision, recall, and F-score of our initial model

PRECISION	RECALL	F-SCORE
70.77	60.53	65.25

Table 4. Precision, Recall, and F-score for each entity type in our initial model

ENTITY TYPE	PRECISION	RECALL	F-SCORE
FILENAME	50.00	40.00	44.44
MALWARE_NAME	60.00	84.00	70.00
VULNERABILITY	57.14	100.00	72.73
OPERATING_SYSTEM	71.43	71.43	71.43
SOFTWARE_NAME	90.00	69.23	78.26
VERSION_TAG	25.00	33.33	28.57
FILEPATH	0.00	0.00	0.00
PROTOCOL	33.33	10.00	15.48
THREAT_ACTOR	100.00	100.00	100.00
CAMPAIGN	50.00	33.33	40.00
MALWARE_TYPE	0.00	0.00	0.00

3.3.1. REPRESENTATION OF ENTITY TYPES

Table 2 displays the number of annotations for each entity type annotated during the first round of annotations. The left half of the table includes cybersecurity domain entity types that we defined. The right half of the table includes general spaCy types that the out of the box model is trained to recognize. The general types are pre-populated during the annotation engine instantiation. We modified this provided for the second round of annotations. More information on modifications to this list is described in Section 3.3.3.

We observe that the occurrence of entity types in a document can impact the number of annotations for certain types, unbalancing the data distribution when training. For example, we noticed limited occurrence of spaCy general types in the CTI corpus. However, certain general types like *ORG* and *DATE* have higher frequency. In terms of cybersecurity specific types, we notice that *Malware_Name* has high frequency, as most of the articles describe technical details of APT groups and malware types. Since some of the less frequent types did not have enough annotations to train the Entity Recognizer model, the F score lowered as a result.

3.3.2. INTER-ANNOTATOR AGREEMENT

In addition, there was high disagreement between annotators. Taking the maximum of each group's annotations, there was a total number of 1755 annotations. However only 781 annotations were accepted, based on the Inter-Annotator Agreement scores. One observation to explain the disagreement is multiple definitions of entity types, such as *Product* and *Software Name*. Broadly, there are many variations for the methods in which one could annotate these entities. For example, the annotators often defined software libraries as products. In addition, discrepancies such as *Windows* versus *Windows OS*, can drastically change the distribution of the data. A similar conflation happened between *Software_Name* and *Tool*. Another issue was in entities that could be classified as multiple types. Consider the following example: "Microsoft Word". An annotator could recognize is Microsoft an *ORG* and Word the *Software_name*, or could merge the terms together as the name of a single software type. Lastly, the final potential issue was syntactical: if a process was called *RunGame()*, some annotators would include the parentheses and some not. The annotator disagreement issues also contributed to some entity types lacking annotations in the final dataset.

3.3.3. IMPROVING HUMAN ANNOTATION STUDY

To improve our dataset we are undergoing a second round of annotations, with comprehensive changes such as the introduction of the rule-based entity pre-population, and further annotator training. Another change we made was modifying the initial list of entity types. For example, we removed the type *Tool* because it was difficult to differentiate it from *Software*, and in many cases the types were semantically, the same. We deemed *Process* to be more general and nebulous to annotate well, so we redefined it into *Function* as it was more agreeable to multiple sub domain definitions. We also redefined *Filepath* to just *Path*, and added the *File_Extension* type. To further help mitigate annotator disagreement, we provided more in-depth training and documentation to the annotators.

Our strategy for mitigating the lack of annotations for certain types is to implement rule-based recognition for these types so they require fewer annotations to recognize at an acceptable level. We especially target entities with low annotation numbers for this, such as *Attack_Type* or *Port*. These rule-based recognizers also correspond to entities that had very low F-scores in 4. Entity types such as *Filepath*, *Malware_type*, *Protocol*, and others which had low F-scores are those that we are implementing rule-based recognizers for in the updated pipeline, whereas entity types such as *Malware_Name*, *Software_Name*, and others perform fairly well without these and we believe will perform even better with more annotations.

These rule-based recognizers were also applied during the second round of annotation, which also helps reduce annotator confusion as the entities are already labeled. Our final measure was to increase the size of the annotation set to 150 paragraphs of text for each annotator to ensure each entity receives a consistent set of annotations.

3.4. Conclusion

We described a preliminary framework for improving entity recognition and linking for the cybersecurity domain. We created a corpus of cyber threat intelligence, developed

a human annotator study to create high quality and shareable cybersecurity entity recognition datasets, tested spaCy framework capabilities to improve automatic entity extraction, and identified custom sets of Wikidata types and properties to support the cybersecurity domain.

In future work, we will integrate our Wikidata entity linker into the pipeline, add a *coreference module* to link pronominal and nominal mentions to entities, train a relation-extraction module for the domain, and adapt our earlier work to extract information on cybersecurity events (Satyapanich et al., 2020). We also plan to develop a system to automatically do periodic web scraping and processing of the new text. The cybersecurity entities, relations, and events will be used to populate and extend existing CKGs (Piplai et al., 2020; Mitra et al., 2022).

Acknowledgements

This research was supported by grants from NSA and the National Science Foundation (No. 2114892).

References

Agarwal, O., Ge, H., Shakeri, S., and Al-Rfou, R. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688*, 2020.

Alam, M. T., Bhusal, D., Park, Y., and Rastogi, N. CyNER: A Python Library for Cybersecurity Named Entity Recognition. *arXiv:2204.05754 [cs.CR]*, April 2022.

Arnold, N., Ebrahimi, M., Zhang, N., Lazarine, B., Patton, M., Chen, H., and Samtani, S. Dark-net ecosystem cyber-threat intelligence (CTI) tool. In *Int. Conf. on Intelligence and Security Informatics*, pp. 92–97. IEEE, 2019.

Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web. *Scientific american*, 284(5):34–43, 2001.

Bridges, R., Jones, C., Iannacone, M., Testa, K., and Goodall, J. Automatic Labeling for Entity Extraction in Cyber Security. *arXiv preprint arXiv:1308.4941*, 2014.

Connolly, J., Davidson, M., and Schmidt, C. The trusted automated exchange of indicator information (taxii). *The MITRE Corporation*, pp. 1–20, 2014.

Gao, C., Zhang, X., Han, M., and Liu, H. A review on cyber security named entity recognition. *Frontiers of Information Technology & Electronic Engineering*, 2021.

Georgescu, T.-M., Iancu, B., Zamfiroiu, A., Doinea, M., Boja, C. E., and Cartas, C. A survey on named entity recognition solutions applied for cybersecurity-related text processing. In *5th Int. Congress on Information and Communication Technology*, pp. 316–325, 2021.

Honnibal, M. Embed, encode, attend, predict: The new deep learning formula for state-of-the-art nlp models. *Explosion AI*, 2016.

Honnibal, M., Montani, I., Van Landeghem, S., and Boyd, A. spaCy: Industrial-strength natural language processing in Python. https://spacy.io/, 2020.

Joshi, A., Lal, R., Finin, T., and Joshi, A. Extracting cybersecurity related linked data from text. In *7th Int. Conf. on Semantic Computing*, pp. 252–259. IEEE, 2013.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., and Auer, S. DBpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

Loper, E. and Bird, S. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

Mitra, S., Piplai, A., Mittal, S., and Joshi, A. Combating Fake Cyber Threat Intelligence using Provenance in Cybersecurity Knowledge Graphs, January 2022.

Mittal, S., Das, P., Mulwad, V., Joshi, A., and Finin, T. Cybertwitter: Using twitter to generate alerts for cybersecurity threats and vulnerabilities. In *Int. Conf. on Advances in Social Networks Analysis and Mining*. IEEE, 2016.

Mittal, S., Joshi, A., and Finin, T. Cyber-all-intel: An AI for security related threat intelligence. *arXiv:1905.02895*, 2019.

Mulwad, V., Li, W., Joshi, A., Finin, T., and Viswanathan, K. Extracting information about security vulnerabilities from web text. In *Int. Joint Conf. on Web Intelligence and Intelligent Agent Technology - Workshops*. IEEE, 2011.

Oasis. Oasis cyber threat intelligence (CTI). http://oasis-open.github.io/cti-documentation, 2021.

Pingle, A., Piplai, A., Mittal, S., Joshi, A., Holt, J., and Zak, R. RelExt: Relation extraction using deep learning approaches for cybersecurity knowledge graph improvement. *IEEE/ACM Int. Conf. on Advances in Social Networks Analysis and Mining*, 2019.

Piplai, A., Mittal, S., Joshi, A., Finin, T., Holt, J., and Zak, R. Creating cybersecurity knowledge graphs from malware after action reports. *IEEE Access*, 2020.

Pradhan, S. S., Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., and Weischedel, R. Ontonotes: A unified relational semantic representation. In *Int. Conf. on Semantic Computing*, 2007.

Ranade, P., Piplai, A., Joshi, A., and Finin, T. CyBERT: Contextualized embeddings for the cybersecurity domain. In *Int. Conf. on Big Data*, pp. 3334–3342. IEEE, 2021.

Samtani, S., Abate, M., Benjamin, V., and Li, W. Cybersecurity as an industry: A cyber threat intelligence perspective. *The Palgrave Handbook of International Cybercrime and Cyberdeviance*, pp. 135–154, 2020a.

Samtani, S., Zhu, H., and Chen, H. Proactively identifying emerging hacker threats from the dark web: A diachronic graph embedding framework (D-GEF). *Transactions on Privacy and Security*, 23(4):1–33, 2020b.

Satyapanich, T., Ferraro, F., and Finin, T. CASIE: extracting cybersecurity event information from text. In *34th AAAI Conference on Artificial Intelligence*, 2020.

Song, C. H., Lawrie, D., Finin, T., and Mayfield, J. Gazetteer Generation for Neural Named Entity Recognition. In *33rd Int. FLAIRS Conf.* AAAI, May 2020.

Tounsi, W. and Rais, H. A survey on technical threat intelligence in the age of sophisticated cyber attacks. *Computers & security*, 72:212–233, 2018.

Vrandečić, D. and Krötzsch, M. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57 (10):78–85, 2014.

Wagner, C., Dulaunoy, A., Wagener, G., and Iklody, A. MISP: The design and implementation of a collaborative threat intelligence sharing platform. In *Workshop on Information Sharing and Collaborative Security*. ACM, 2016.

Wang, L., Zhang, Y., and Liu, T. A deep learning approach for question answering over knowledge base. In *Natural Language Understanding and Intelligent Applications*, pp. 885–892. Springer, 2016.

Wirkuttis, N. and Klein, H. Artificial intelligence in cybersecurity. *Cyber, Intelligence, and Security*, 1(1), 2017.

Wise, C., Ioannidis, V., Calvo, M., Song, X., Price, G., Kulkarni, N., Brand, R., Bhatia, P., and Karypis, G. Covid-19 knowledge graph: accelerating information retrieval and discovery for scientific literature. *arXiv:2007.12731*, 2020.