information

Article

Building Knowledge Graphs from Unstructured Texts: Applications and Impact Analyses in Cybersecurity Education

Garima Agrawal [1,*], Yuli Deng [1], Jongchan Park [2], Huan Liu [1] and Ying-Chih Chen [2]

[1] School of Computing and Augmented Intelligence, Arizona State University, Tempe, AZ 85281, USA
[2] Mary Lou Fulton Teachers College, Arizona State University, Tempe, AZ 85281, USA
* Correspondence: garima.agrawal@asu.edu

Abstract: Knowledge graphs gained popularity in recent years and have been useful for concept visualization and contextual information retrieval in various applications. However, constructing a knowledge graph by scraping long and complex unstructured texts for a new domain in the absence of a well-defined ontology or an existing labeled entity-relation dataset is difficult. Domains such as cybersecurity education can harness knowledge graphs to create a student-focused interactive and learning environment to teach cybersecurity. Learning cybersecurity involves gaining the knowledge of different attack and defense techniques, system setup and solving multi-facet complex real-world challenges that demand adaptive learning strategies and cognitive engagement. However, there are no standard datasets for the cybersecurity education domain. In this research work, we present a bottom-up approach to curate entity-relation pairs and construct knowledge graphs and question-answering models for cybersecurity education. To evaluate the impact of our new learning paradigm, we conducted surveys and interviews with students after each project to find the usefulness of bot and the knowledge graphs. Our results show that students found these tools informative for learning the core concepts and they used knowledge graphs as a visual reference to cross check the progress that helped them complete the project tasks.

Keywords: knowledge graph; knowledge graph question answering systems (KGQA); natural language processing (NLP); ontology; cybersecurity

Citation: Agrawal, G.; Deng, Y.; Park, J.; Liu, H.; Chen, Y.-C. Building Knowledge Graphs from Unstructured Texts: Applications and Impact Analyses in Cybersecurity Education. *Information* **2022**, *13*, 526. https://doi.org/10.3390/info13110526

Academic Editor: Ryutaro Ichise

Received: 27 September 2022
Accepted: 2 November 2022
Published: 4 November 2022

1. Introduction

The ability to represent knowledge, visualize, manage and retrieve useful information has made knowledge graphs more popular than other traditional knowledge-oriented information management and mining systems. After Google coined the term 'Knowledge Graph' (KG) in 2012 [1], mainly for the purpose of web-based semantic search, knowledge graphs have undergone many technical advancements and broader applications both in industry and academia. Knowledge graphs are basically an extension of semantic nets [2], which represent knowledge in the form of interconnected nodes and arcs, where nodes represent 'objects' or 'concepts' and the arcs or edges represent the interaction or relations between them. The semantic nets lacked formal syntax and semantics, whereas knowledge graphs overcame the limitations of these methods by defining a structured schema and a formal way of interpreting the graphs. Knowledge graphs have an entity-centric view; they not only focus on the structured representation of semantic knowledge but also on how the entities are linked, interpreted and disambiguated. This helps in checking the correctness, connectedness and consistency of information. Knowledge graphs provide flexibility for the user in adding meaningful information or 'explicit knowledge' in the form of entity attributes and relational labels. The convention to define the scope and semantic conditions of entities and relations in a given domain is called an *ontology*. It is crucial to have a well-defined ontology for the construction of a domain-specific knowledge graph as the ontology enables domain entailment and drives the behavior of entities and

relations as per the domain's requirement. The ontology essentially provides a mapping between the domain and the graph. It defines a set of certain rules and constraints that are specific to a domain and helps the user make inferences while following the underlying schema [3]. Other than the general purpose knowledge graphs for Google searches, most knowledge graphs are domain-specific and conform to the underlying ontology. There have been efforts to automatically generate knowledge graphs using labeled datasets or by extracting information from Wikipedia or other web data but the graphs generated using those methods are incomplete and inconsistent [4]. Constructing the knowledge graph for a new domain in the absence of ontology or a standard dataset is challenging.

It is overwhelming for a human expert to start designing the schema from scratch. There have been efforts to perform the generalization of ontology instances for similar domains [5] or to build ontologies for generalized knowledge [6], but these methods are not successful for specialized domains. It is practically impossible to build a schema that can fit all domain-specific requirements and constraints, and it is able to answer all kind of queries from different industries such as medicine, product manufacturing, urban planning, cybersecurity, social media, education, road planning, etc. [7].

The deep learning methods can still be used to automatically construct knowledge graphs from data on the web or use the standard training datasets with labeled entities, if those are available for a given domain [8–10]. However, most domains do not have annotated datasets and if the datasets are present in some cases, the automatic knowledge graph construction methods lack a comprehensive view of the domain. In such cases, the quality of the knowledge graphs and their usefulness for an application becomes questionable [11]. For an application to effectively consume and encode the knowledge from the knowledge graphs for different information processing tasks such as search, recommendation, question answering, generating summary, etc., it may be important to provide extensive background domain knowledge using the standard schema and vocabulary [12]. This has always been a research dilemma if knowledge graphs should be created exclusively using automated machine learning based methods or should be curated by human experts. Essentially, a combination of both methods would be more appropriate [13].

The knowledge graphs can be used by an application for downstream tasks only when they are modeled as per the domain-specific requirements defined in the ontology. The ontology's development is more reliable when performed by human domain experts as it demands an in-depth understanding of domain and requires attention to detail. It is one of the most crucial steps while architecting a knowledge graph. The efficiency of the ontology development is also dependent on the knowledge acquisition process. If the data are stored in a structured relational database and data sources are well-integrated, then it is relatively easier for the domain expert to extract the entities and relations from the given relational schema. Unfortunately, most domain knowledge is possessed by the human expert or it resides in multiple data sources in the form of digitized or printed unstructured texts, product manuals, catalogs, web portals, legacy information systems, etc. It can be quite challenging to first integrate a large amount of data from different sources and then to visualize it to identify the key entities and relations [14,15].

Knowledge Graphs for Cybersecurity Education

In this research work, we explored the challenges of constructing the knowledge graph from unstructured texts for education in cybersecurity. The knowledge graphs have been effectively used in education [16] and cybersecurity is one of the cutting-edge domains that should exploit the potential of knowledge graphs to create an interactive and adaptive learning environment to teach cybersecurity professional skills.

To be able to use and deploy state-of-the-art tools and technologies in cybersecurity, it is crucial to prepare a cybersecurity specialist workforce. The students or cybersecurity professionals are not only required to master cybersecurity concepts but also learn critical problem-solving skills based on different attack scenarios. The students need to focus on

learning the fundamentals of web application security with a solid foundation of offensive and defensive skills which can be developed only through hands-on experience. We need instructional strategies that can help students not only learn the complex nature of attacks and the techniques for detecting common vulnerabilities and defense mechanisms but also give a holistic view of real-world scenarios and promote problem-solving skills for cybersecurity.

The knowledge graphs are best suited for these purposes as they provide concept visualization and promote cognitive engagement [17,18]. The representation of cybersecurity concepts and their interaction in the form of knowledge graphs can be further consumed for downstream tasks such as designing problem-based learning paradigms and creating learning recommendations and interactive question answering systems for students. Unfortunately, there are not any well-defined ontology or labeled entity-relation datasets for education in cybersecurity.

In this work, the first challenge was to define an ontology specific to this domain. We observed that the data are present in multiple forms in the course's portal such as instructor-led class videos, lecture notes, projects, lab manuals, setup guides, assignments, quiz, etc., and there are also external learning sources such as Wiki pages, YouTube, capture the flag (CTF) and other hacking challenges. The data were in the form of unstructured complex course materials, and it was overwhelming for a human expert to manually curate key entities and relations from scratch. Thus, we used a bottom-up approach using machine learning and domain expert knowledge to curate the entity–relation pairs. We first applied natural language processing based named-entity methods to extract the entities and generated the preliminary graphs for the first few lab manuals [19]. The visual representation made it easier to identify key entities and relations. We then used our domain knowledge to add the attributes and constraints and define the ontology. Using our ontology framework, we created a graph database and constructed the knowledge graphs. In this work, we suggest an approach to use a combination of both machine and human-curated knowledge to build the ontology.

The second challenge was how to parse a large amount of text in other learning resources to identify the sentences with the key entities. To avoid the tedious annotation work, we wrote a custom entity-matcher in Python to extract the key entities from the remaining text.

Another challenge was how to consume the knowledge graphs and use them for student learning. In order to visualize the concepts, we built a web-based UI. To break down project tasks, we presented sub-graphs for each workflow. We then built a question-answering chatbot as a self-learning tool for students. We used an SVM (support vector mnachine) classifier for intent classification and trained the model on questions that covered key concepts represented by the ontology. We added a chat interface on our web-UI and linked it with the course portal.

To validate the proposed pipeline, we tested our method on another domain called cloud computing and showed that our approach to constructing knowledge graphs, in the absence of standard dataset or ontology, can be applied to any new domain.

To evaluate the effectiveness of the knowledge graphs, we provided the graphs as a learning aid to the graduate-level students while they were working on projects. The results from the surveys and interviews conducted at the end of each project showed that the students found these tools very helpful, and knowledge graphs served as a visual reference in completing their tasks.

The rest of the paper is organized as follows. We discuss the related work in Section 2. Our methodology and proposed framework are described in Section 3. The evaluation and survey results are presented in Section 4. Finally, we conclude the paper in Section 5.

2. Related Work

2.1. Ontology Development

The ontology development and knowledge graph construction is more crucial and challenging when data are available in form of unstructured texts [20]. Wu et al. [4] illustrated the challenges of using different automated techniques in their report for building knowledge graphs automatically from unstructured texts in a specific domain. Zhao et al. [21] constructed the industrial knowledge graphs for unstructured Chinese text learning. Knowledge graphs were generated by extracting information from product specifications to answer the customers questions about the product in e-commerce [22]. AgriKG [23] built an agriculture knowledge graph in Chinese to support agricultural applications. Most works support transforming unstructured texts to a formal representation or using a combination of domain knowledge and automated methods [24–27]. There is a trade-off where the ontology development and knowledge graph construction methods need to strike a balance between the machine vs. human effort [28–30].

2.2. Knowledge Graph Construction in Cybersecurity Domain

Security is an important aspect both in software development and networking. Most studies that recommend using knowledge graphs in cybersecurity deals with modeling systems and monitoring tools to detect cyber threats, vulnerabilities and security design flaws, etc. [31–36]. Cybersecurity is a dynamic stream and a very specialized skill to develop. With the rise in cyber crimes, it is extremely important to focus on training cyber professionals. Cybersecurity-related challenges involve multi-facet complex real-world problems that demand adaptive learning with tangible feedback and assessments to motivate students in using critical thinking to solve the given tasks. The student should gain experience in detecting common vulnerabilities and attacks, using exploitation techniques and strategies to protect the applications and systems, etc. The learning strategy needs to be designed in a way that can breakdown complex tasks and concepts into simpler representations and engage students in active learning.

Knowledge graphs have been used effectively in education to enhance learning experiences, and there has been a rise in the demand of knowledge graphs in the education domain. Knowledge graphs have been used in improving the knowledge management process at universities [37]. Chen et al. proposed KnowEdu [38] and K12Edu [39] to automatically construct knowledge graphs for the education domain based on the standard curriculum for K–12 education. Aliyu et al. [40] used them to model university course-management systems. Qin et al. [16] showed improvement in teaching quality by using knowledge graphs.

Deng et al. [41,42] are pioneers working in the field of cybersecurity education and suggested the use of knowledge graphs as guidance for hands-on labs. However, earlier works focused on finding the similarity between different cybersecurity concepts by using word embeddings to create graphs for similar concepts found on the web [43]. In this work, our objective is to develop an end-to-end architecture of knowledge graph construction for cybersecurity education. We present a knowledge acquisition and construction framework to extract cybersecurity-related entities and relations for ontology development from unstructured course material. We then developed a UI to visualize the graphs and knowledge consumption layer to answer student queries using an interactive chatbot. We also show that this approach can be used to construct the ontology or knowledge graphs for a greenfield project in any other domain.

3. Methods

In this section, we present our approach for building the knowledge graphs (KG) for cybersecurity education from unstructured course materials. The purpose of this research work is two-fold:

- From the student perspective, we have the following:
  1. *Concept Visualization*: The first objective was to provide a visual concept graph for students to learn the key concepts of a complex course such as cybersecurity. The conceptual graphs help in breaking down task instructions, thereby allowing the students to visually analyze and implement the project's challenges.
  2. *Question Answering*: As a downstream task, we built an interactive chatbot for the students to promote self-learning. The students are able to query the system for frequently asked questions on the lab system's setup, concepts, projects, etc. We trained the bot using an SVM model for intent-classification based on the questions from the key entities that were identified in our ontology. By using the bot, we also had the opportunity to collect the logs of more questions and paraphrases from the students. This allowed us to retrain the existing model. In the future, we aim to build a dataset to train a question-answering model on cybersecurity education.
  3. *Student Feedback*: As a part of our work, we wanted to find the impact of using knowledge graphs in teaching cybersecurity courses. We took feedback from students via surveys and interviews and analyzed whether visual graphs and question-answering chatbot served as effective learning aids for students. The feedback from students also helped in improving our knowledge graph consumption tools.
- From the research perspective, we have the following:
  1. *Ontology Framework for Cybersecurity Education*: We present a bottom-up approach for identifying the key entities and relations from unstructured course material for cybersecurity education. We developed an ontology framework that can be very well adapted to any ontology standards and query languages across industry and academia for constructing knowledge graphs for cybersecurity education. We also show that this approach can also be used by other domains to build ontology from scratch.
  2. *Research Potential*: With the increase in demand for cybersecurity professionals, there is a need to build effective learning tools. In spite of the fact that acquiring cybersecurity skills is challenging, there is not much research traction in this field. Knowledge graphs can be extremely helpful in creating interactive and adaptive learning required for cybersecurity. However, there are no data models with standard key entity–relation pairs or labeled datasets, so there is no starting point for building knowledge graphs for cybersecurity education. Our work is a gateway to showcase the research potential and implementation opportunities in this area.

In the following subsections, we are going to present the knowledge graph architecture's phases and the implementation of each component. To construct an architectural structure of the knowledge graph not only comprises modeling the data and constructing graphs but also involves the ability to store and consume knowledge. The block diagram in Figure 1 represents all three phases used in our method. The first phase is called knowledge acquisition and integration. It includes data modeling and ontology development. The second phase is knowledge storage and the final phase is the knowledge consumption layer, which provides the applications such as visualization, semantic reasoning, question answering, etc., to the end-user. In our case, the application is viewed via the web-based UI and the end users are the students. The processes used in each phase are explained further.

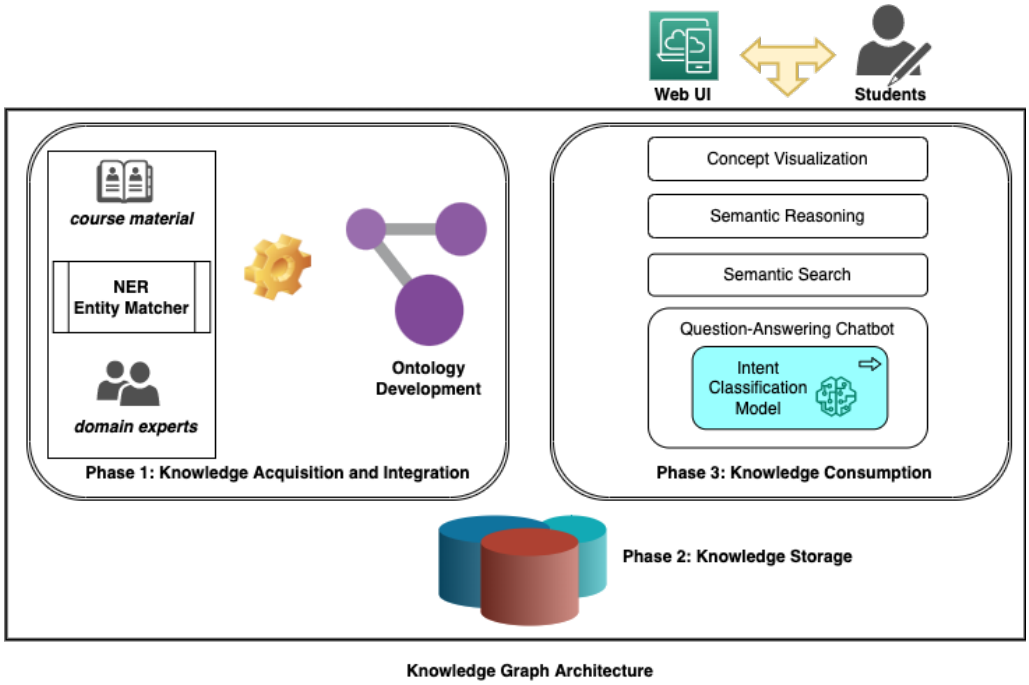


Figure 1. Architecture for Knowledge Graph as explained in Section 3.

3.1. Phase 1: Knowledge Acquisition

The first phase of building the knowledge graphs is to acquire the knowledge. It is one of the most important steps to know and decide what kind of knowledge we want to capture and represent in the knowledge graph, especially for visual representation and other downstream tasks. First, we need to extract key entities and relations. Since we do not have the standard list of entity–relation pairs for cybersecurity, we used a bottom-up approach to first extract raw entities and relationships from the data and then used domain knowledge to design the schema for cybersecurity education. The data modelling steps are given below.

3.1.1. Data Collection

The data were available from educational course materials in the form of course syllabus, course videos, PowerPoint slides, lecture notes, project lab manuals, quizzes and exam questions. We considered only the textual data from the lab manuals for this work. It is not possible to scrape the entire data at once. We prioritized our work and decided to start with the first three lab manuals. Each lab manual is approximately 15–20 pages long. The first manual is about using the network scanning tool called NMAP for network diagnostics and testing. The second manual is for setting up a firewall and using NMAP for penetration testing. The third project was on setting up the Snort tool for network intrusion detection. The manuals are written in standard academic English to explain the concepts, commands and instructions for implementing various lab tasks. To illustrate our approach, we shall be using the running example from the third project on "Network Intrusion Detection using Snort" for the rest of this paper.

3.1.2. Entity and Relation Extraction

Each project manual is a large volume of unstructured texts containing task descriptions and complex instructions. It was overwhelming for a domain expert to go through the course material and to identify key entities and relations. So we used the bottom-up approach by applying natural-language-processing-based named-entity recognition (NER) methods to extract raw entities and relations. The NER method relies on parts of speech (POS) tagging, which extracts the subject and object as entities and the verb or the root of a

sentence as a relation. We performed the following steps for preprocessing the text and building the preliminary visual representation of data.

- ***Sentence Segmentation***: The project instruction manual was split into sentences using a simple python script. Sentences with one subject and one object are easier to extract. For example, the sentence, "Snort can detect attacks" has one subject (*Snort*) and one object (*attacks*). Some sentences were paraphrased such that they had only one subject and object.
- ***Parts of Speech (POS) Tagging and Dependency Parsing***: In the previous example, single-word entities can be extracted easily from a sentence as nouns and proper nouns using part-of-speech (POS) tagging. However, POS tags are not enough for the entities that span multiple words. We used spaCy-based rule-matching methods to parse the dependency tree of the sentence and defined the semantic relationship. We defined a set of rules for the syntax and other grammatical properties to extract meaningful information. The modifiers, prepositions, compound words, and prefixes were considered as the dependencies to be extracted with the subject and objects so that the domain expert could obtain some meaningful information.
  For example, in the sentence, "Snort lab use syslog remote for logging", the parser will now extract the subject as *"Snort lab"* and the object as *"syslog remote for logging"*. Similarly, in the sentence, "Students will write their own IDS rules", the parser will now extract the subject as *"Students"* and the object as *"own IDS rules"*.
- ***Relation Extraction (RE)***: The next task is to extract the relations to connect the entities. POS was used to extract the root or verb of the sentence and tagged it as a relation. We defined the pattern by writing a dependency rule to match the adjectives, adverbs, and auxiliary tokens, etc., while parsing the root.
  For example, for the sentence "Snort can output tcpdump pcap", the relation extracted will be, *'can output'*.
- ***Temporary Triples***: The subject and object extracted from the sentence were stored as entity pairs where the head entity is the subject and the tail entity is the object. The relation extracted from the sentence is the edge label. The triples were stored in the form of a "entity-relation-entity" triple in a csv file. The entities and relations extracted at this stage contain noise and redundant information. The triples thus created are temporary only to provide a visual representation.
- ***Preliminary Visual Representation***: Using the entity and relations extracted from the subject–object pairs and the predicates in the above step, we can now build a visual representation from the unstructured text in the form of a directed labeled graph. One of the sub-graphs on the lab requirements for setting up "Snort" is shown in Figure 2.

We call these graphs preliminary since they are created by extracting raw triples from unstructured text. At this stage, these graphs cannot be consumed directly as the number of entities and relations are ambiguous and too large. Around fifty relations were extracted as roots or verbs and around 300 entities were extracted as subjects and objects of sentences from a single 15-page lab document. Many of the extracted entities and relations were redundant or were only general-purpose words that are not related to the cybersecurity domain. The entities can be categorized into different classes, and relations can be reduced to avoid ambiguity and redundancy in the graph, as explained further. There is a need to give a well-defined schema and rules as per the domain. These graphs are not the final knowledge graphs but a step to provide an intermediary visual view of unstructured data. These preliminary graphs represent the textual information and were used by the domain expert to analyze and identify key concepts and their interactions with each other in the form of entity-relation-entity pairs to design the schema for cybersecurity education.

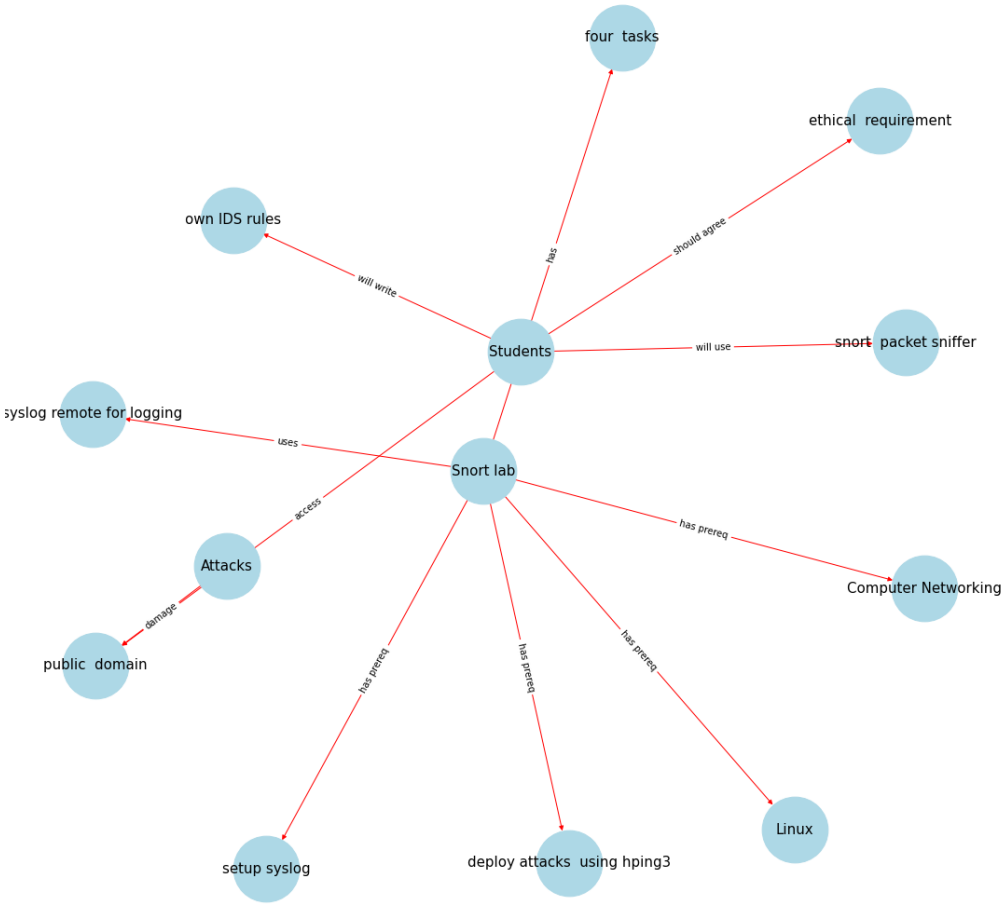


Figure 2. Prelim graph generated from entities and relations extracted using named-entity recognition (NER) as subject–object pairs and predicates in Section 3.1.2. This graphical representation of unstructured text has redundant data, but it is still helpful for the domain expert in visualizing the data and identifying key concepts.

3.1.3. Knowledge Integration

The entities and relations extracted using NER methods are very generic and follow syntactical rules of grammar. NER fails to report ambiguities in the domain-specific entities as it is not entitled with the domain knowledge. It is important to integrate and map named-entities as per their definition in that domain. The same principle applies to relations. The next step in constructing the knowledge graphs is scope resolution, disambiguation and linking entities and relations as per the domain's knowledge.

Entity Linking and Scope Resolution: We first applied lemmatization to obtain the 'base' words for the entities. This step helps in removing redundant entities. There were still multiple entities that were referenced with different names; for example, *Intrusion Detection System* and *IDS* are the same entities in the cybersecurity domain and are often used interchangeably. Similarly, *Configuration Files* and *config files* means the same. Moreover, the thematic scope needs to be resolved. In the sentence, "Snort can detect bad packets", the term *packet* is used for data packets, whereas in the sentence "Packet logging collects the data as file", *Packet log* is rather a feature. So there is clearly a strong need to define an ontology and attach some domain-related meaningful information to the entities.

3.1.4. Ontology Development

As a next step, we used the domain knowledge to develop the ontology framework. The bottom-up approach using NER to discover the ontological knowledge and facts from unstructured data was extremely helpful in learning the ontology. The below subsections

illustrate the key entities, attributes and relations that were identified to encompass the cybersecurity education domain.

The domain experts, working on curating the key entities, should have practiced or demonstrated sufficient knowledge and experience in the subject. In our case, the first author is a graduate researcher in the field of cybersecurity, and the second author is a cybersecurity expert and instructor. He teaches cybersecurity courses at his university to graduate-level students majoring in computer science and cybersecurity.

Key Entities and Attributes for Cybersecurity Education: We observed that there are primarily four distinct categories in which the entities for cybersecurity education can be placed. These categories, as shown in Figure 3, are **concept, application, roles and course**. These categories distinctly divide the sub-areas in the field of cybersecurity education. The "*concept*"-based entities can be further defined as the type **feature, function, attack, vulnerability and technique**. Further, the "*application*" category contains the **tools, system and app**. The different "roles" in cybersecurity education are **student, attacker, ethicalHacker and the user**. Since the domain is related to the cybersecurity course, it is important to define the "course" category with the entity-type, *project* and *course name*. Table 1 gives the list of sample entities under each category.

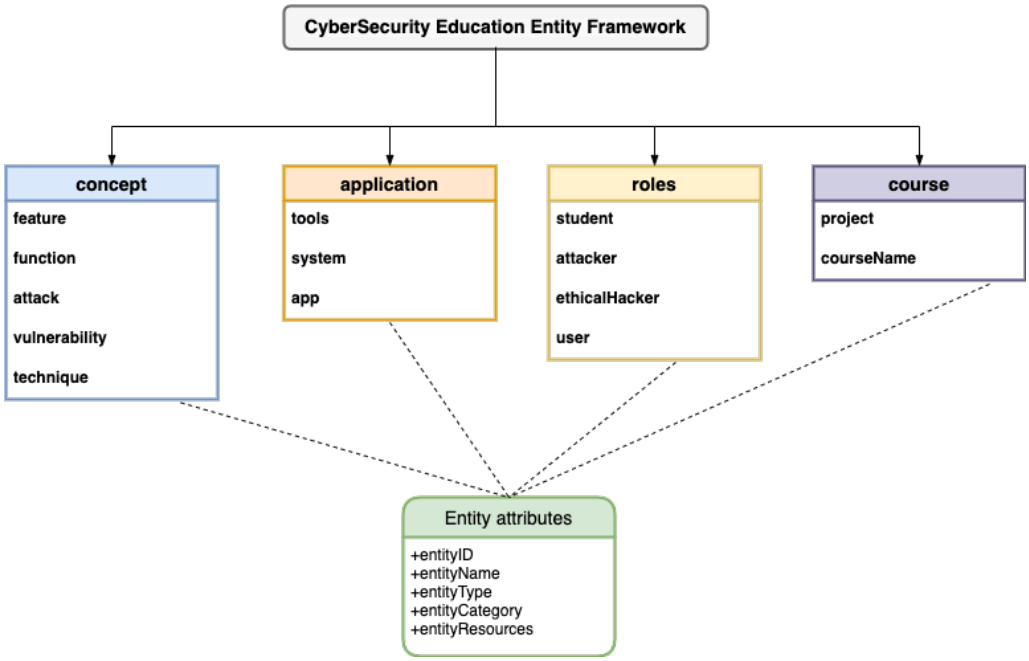


Figure 3. Cybersecurity Education Entity Framework as explained in Section 3.1.4. The figure shows the key entity types identified in the ontology. The green box with entity attributes shows the lists of the common attributes for each entity.

The metadata or the attributes added to entities are entityID, entityName, entityType, entityCategory and entityResources. The entityID is a unique identifier and acts as a primary key to store and access the entities. Other attributes such as name, type and category add semantic information to the entity and were used in entity linking and scope resolution. These are also helpful in semantic search and reasoning for downstream tasks. The entity resource is a Wikipedia link that was added for students to further read about the concept.

Table 1. Example of key entities identified in each category, as explained in Section 3.1.4.

entityName	entityType	entityCategory
private key, cookies, protocol	feature	concept
tcpdump, SHA, hash, xor	function	concept
CSS, Sql Injection, DOS	attack	concept
weak password, poor config	vulnerability	concept
honeypot, risk assessment	technique	concept
burp, snort, wireshark	tools	application
linux, IP, Server	system	application
browser, webapp	app	application
attacker, black hat	attacker	roles
security engg, white hat	ethicalHacker	roles
employee, user	user	roles
student	student	roles
task4, project3	project	course
CSE575, lab-CNS-003	courseName	course

Relations for Cybersecurity Education: There were 50 roots extracted as relations from NER out of which many were redundant or had slight differences varying with different sentence formations. We chose the most common relations and also analyzed the interaction of entities in our ontology with each other using domain knowledge. We finally came up with the following eight relations for this ontology: **has_a, can_analyze, can_expose, has_prereq, can_exploit, can_cause, implements and uses**.

In order to ensure that an adequate amount of information is depicted in the knowledge graphs, we added an attribute called "**action**" to the relations. This is similar to reification in RDF standards or adding a singleton property to relations in property graphs, but since we are proposing only a generalized ontology framework that can be adapted to any ontology standards [44], we called it a relational attribute: "action". The purpose of "action" is to add more information about the activity that students need to perform for the task. This knowledge was derived from the preliminary visual graphs generated from NER. The edge in the final graphs shows the "relation" as the top label and "action" at the bottom. If there are no additional actions, we mark it as NULL in the database, and it is not shown visually in the graph.

Figure 4 shows two different samples of triples with the entity and relational attributes depicted in the nodes and edges. Figure 4a shows the entity "Snort" and its attributes. The edge has a label, "has_a", and this is the relation between head entity "Snort" and tail entity "PacketDecoder". In Figure 4b, the "action" attribute is added at the bottom of the edge to provide additional information, which means that students need to detect the "SmurfAttack" while implementing Task4 of the project.

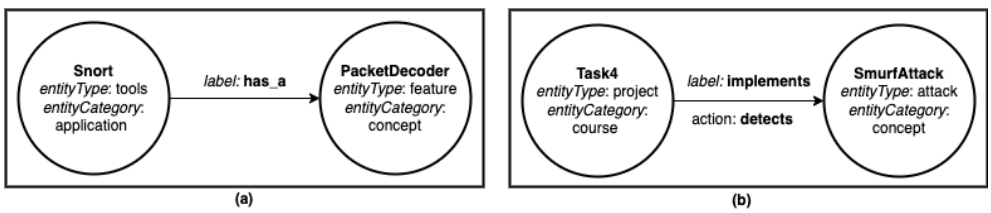


Figure 4. Examples of entity-relation-entity triples in our ontology, as given in Section 3.1.4. (**a**) The first figure represents the entity "Snort" and its attributes, entity ID, type as "tools" and category as "application". The "has_a" relation follows with another entity "PacketDecoder" which is of the type "feature" under the category "concept". (**b**) The second figure shows the relational attribute "action" as "detects", to show the additional activity performed during the interaction between the project, "Task4" and attack type entity "SmurfAttack".

Constraints and Rules: Ontology also provides certain rules and constraints that the entities in schema should follow as per the domain. We defined unique constraints on the

primary key for entityID. For any other data definition (DDL) operations such as insertion, cloning, merging or the removal of nodes on graph, we defined schema validation rules using a property graph. Figure 5 shows the schema for cybersecurity education. Figure 5a shows the list of nodes that the graph instance is allowed to create. These nodes are the same as the types of entities that have been identified in the ontology. Figure 5b shows the tuples or the edges as defined by our ontology. For example, the schema-edge (function, can_expose, attack) here represents all such tuples in which the entity of type "function" has a relation "can_expose" with the entity of type "attack". Thus, the tuple ('Snort Rules", can_expose and "Land Attack") is a valid triple. The graph instance following this schema will not be able to create any other tuple outside this definition. However, this schema will continue to evolve as we discover more entities and relations in the domain.

To avoid any inconsistency or noisy data in graphs, the property graphs allow us to define the schema and validate it formally through the concept of homomorphism. The homomorphism $h : G \rightarrow S$ basically means that the "graph", G, that respects the given "schema", S, in each node and edge in G is an instance of S, and they follow mandatory properties in S, and the edges in S constrain which edges can exist in G [45]. We used a Python library called Regraph [46] to create schema-aware graph objects. Regraph provides utilities for rewriting graphs with attributes based on the sesqui-pushput-rewriting approach [47] for creating rules to apply on the graph objects to maintain consistent data. In the case of any changes in the schema, it allows us to transform the existing graph by reapplying the rules via matching the instances and also restricts any new DDL operation to strictly follow the latest schema.

Schema Nodes
project
courseName
student
user
ethicalHacker
attacker
app
system
tools
technique
vulnerability
attack
function
feature

(a)

Schema Edges or Triples as per Ontology	
(courseName, has_a, project),	(attacker, can_exploit, vulnerability),
(project, has_prereq, system),	(attacker, can_exploit, feature),
(project, has_prereq, concept),	(ethicalHacker, can_analyze, vulnerability),
(system, can_expose, attack),	(ethicalHacker, uses, tools),
(system, can_expose, vulnerability),	(ethicalHacker, can_exploit, app),
(app, has_a, feature),	(ethicalHacker, implements, technique),
(tools, has_a, function),	(user, uses, system),
(tools, can_analyze, function	(user, can_expose, vulnerability),
(tools, can_analyze, apps),	(student, uses, system),
(tools, can_analyze, vulnerability),	(student, can_analyze, feature),
(tools, implements, tools),	(student, implements, function),
(function, can_expose, attack),	(student, implements, technique),
(function, has_a, feature),	(student, uses, tools)
(feature, can_cause, attack),	

(b)

Figure 5. Cybersecurity education ontology schema. (**a**) The first figure represents the valid nodes in the schema (**b**) The second figure shows the schema edges or triples as per the ontology. Any new DDL operation such as insertion, cloning, merging or the removal of nodes on graph is schema-aware and should follow these tuples as per schema validation rules.

3.1.5. Entity Matcher

Once the baseline definition of our schema containing various entity types, attributes, relations and constraints is established, the next task is to extract entities from remaining course materials. One of the commonly used methods is to manually annotate the data and label the entities. However, it is an extremely time-consuming and expensive task. To avoid the annotation effort, we wrote a Python script for an entity matcher and created our custom knowledge base using the *spacy* library. Although the out-of-the-box *spacy* is limited to recognizing only the general named-entities such as person, organisation (ORG), geopolitical entity (GPE) for location, products, money, date, etc., *spacy* also allows creating custom knowledge base (KB) using the entity "PhraseMatcher" function. In our custom KB, we added the semantics by defining the entity type and corresponding lexical entries for the entities pertaining to cybersecurity as per our ontology. We need to ensure that the corresponding lexical entries for the entities are matched and linked to the same entity. For example, IP and Internet Protocol are the same and should be linked. The entities with their class or types were maintained in a knowledge base (KB) using a yaml file

and the corresponding lexical names, abbreviations and synonyms for those entities are maintained in another yaml file. The entity matcher program extracts all sentences that contain key entities.

To ensure that new entities that are not part of the KB but that are seen in the document are not missed, the entity matcher module was made to match both the custom entities defined in the custom KB as well as the general NER entities. In cases where we find new entities that are related to cybersecurity education, the KB needs to be manually updated and the corresponding entry is made in the lexical file. The lab documents were again scanned manually by human experts to validate if all potential entities have been considered and added to the KB. Using the entity-matcher approach automated the cybersecurity-based entity recognition process and saved a substantial amount of labeling time and effort.

3.2. Phase 2: Knowledge Storage

The knowledge storage layer in the architecture is mainly about being able to store triple data and access it. It is a crucial step as the choice made to store the data might impact the factors such as ease and flexibility to further evolve and add a new type of knowledge. It also affects the future application of knowledge graphs in downstream tasks. There are multiple choices to store the knowledge: Either the traditional relational databases can be used or any of the graph databases such as RDF Triplestore [48], Neo4j [49], DGraph, ArangoDB, etc., are good choices to store the triple data. Moreover, graph query languages such as SPARQL, Cypher [50], SQL/PGQ, GSQL, GraphQL [51], G-core [52] and PGQL [53], etc., can be used to query graphs for semantic search.

We initially used the networkX library in Python to generate the graphs, and the triples were maintained in a csv files, but as the volume of our dataset grew, we decided to create our own DBMS in Neo4j and ported the csv files into the Neo4j DBMS using the Neo4j sandbox environment. The relationships were created as per our ontology schema and we added unique constraints on the primary key and data types. We used the "ReGraph" Python library to write rules to ensure that the insertion, cloning, removal or merge DDL operations on incoming triple data are schema-aware. To persist our data store, we connected the sandbox and ported the data to the local Neo4j Desktop plugin.

We chose Neo4j, as the property graphs are flexible for storing multiple properties for both nodes and edges. Given the nature of our domain, it is likely to add more attributes or properties to enrich our schema and create a dataset to facilitate advanced reasoning tasks. Moreover, property graphs maintain key–value pairs that provide more flexibility for real-world applications. The same key can represent the element in case of an entity being called by multiple names or acronyms. Thus, the key can be used as a unique identifier for that entity, which is "entityID" in our case.

3.3. Phase 3: Knowledge Consumption Layer

Knowledge consumption's purpose is to make knowledge accessible and usable for the users. In our case, the end-consumers are graduate students in our university. One of the main purposes of this work was to provide a visual representation of the project tasks and build a question-answering system for self-learning. If the students are made to access the stored knowledge by writing graph queries, they may not find it useful. The graph query languages require the user to know the exact syntax and schema of the data. As most students are cybersecurity majors, they may not be well-versed with the syntax of graph query languages.

Thus, we built two components for the students in the consumption layer. The first one was to provide a visual view of the knowledge graphs. The second component is a question-answering chatbot that was developed for students to answer course-related questions. The students can type their questions in simple natural language and obtain a related response. For Bot, a support vector machine (SVM) model was trained to provide the response based on intent classification.

3.3.1. View Knowledge Graphs

The knowledge graphs are too large to view and fit in a single frame. Some entities overlap and may be hard to comprehend. To solve this, the knowledge graphs were extracted as per the simple query match on the entity types, *course and project*. We provided the list of tasks as clickable links and the query runs on the backend to fetch the corresponding graph. Figure 6 shows the final knowledge graph that was generated for project3 to detect Land attack using our ontology. We developed a web-based UI to view these graphs and the web page was linked to the course portal so students could easily access it. The web page was written in PHP and JavaScript using the bootstrap framework on an Apache server hosted on the ASU private cloud.

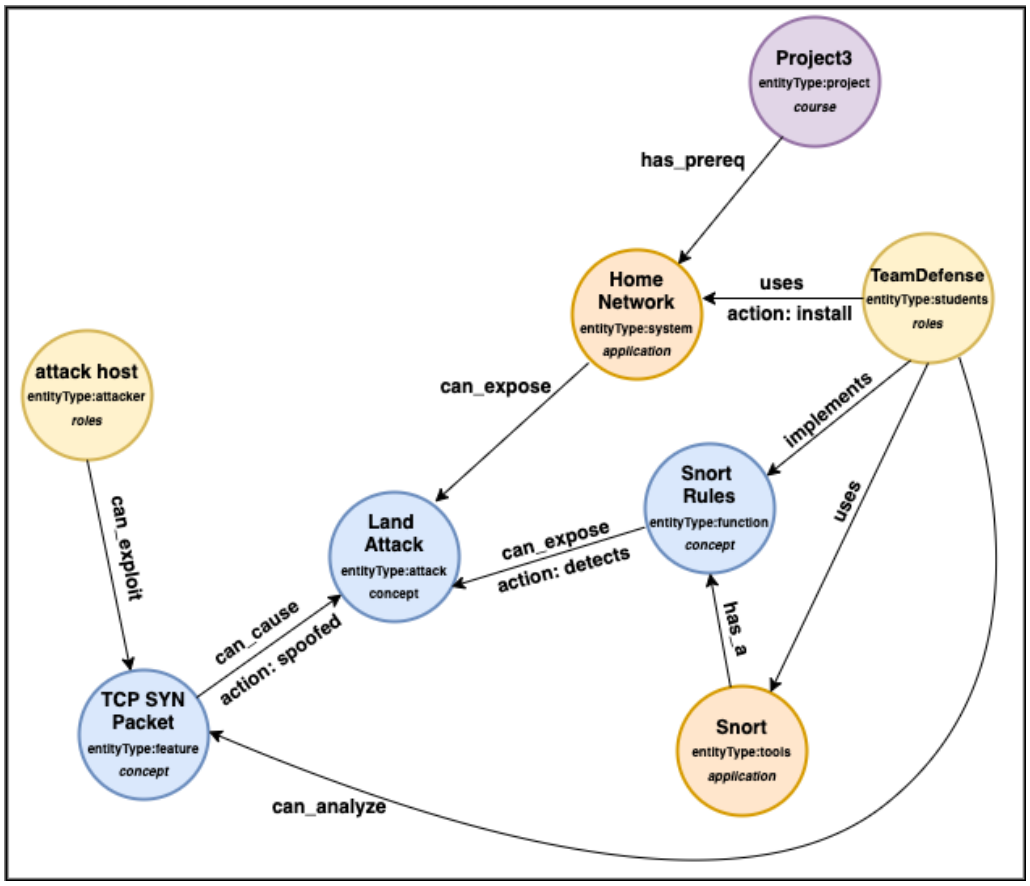


Figure 6. A knowledge graph with entity-relation triples based on our schema to give an overview of lab exercise for using snort rules to detect a land attack.

3.3.2. Chatbot

The final component is the question-answering system, which allows the students to ask questions in natural language and to obtain the relevant answer. The traditional keyword search is not effective as it returns the list of related documents from the knowledge stores. Moreover, the students are expected to use every link to find what they are looking for. It is time consuming and does not provide a good learning experience for students. We automated this process of question answering by building a machine-learning based chatbot that classifies the intent of the questions asked in natural language and provides the answer corresponding to that intent. Figure 7 shows a snapshot of conversations in the chatbot.

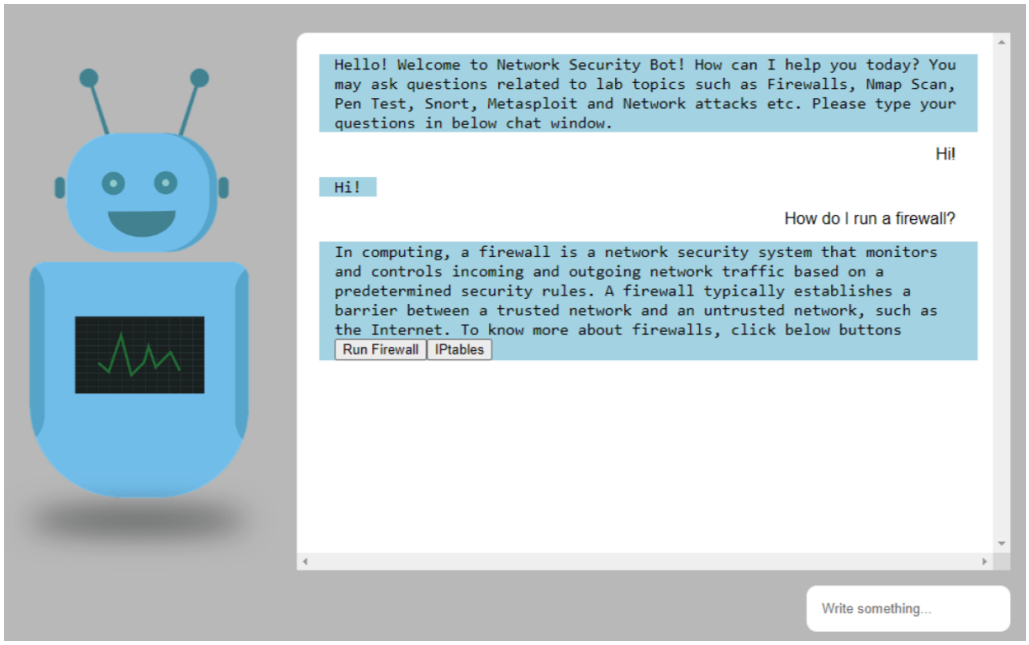


Figure 7. The Chatbot developed for students to answer the cybersecurity course-related question answers. The query is sent as a request to the SVM model, which identifies the intent of the question and returns the class or label. The corresponding response is fetched from the json file and sent to the user. To address ambiguity and avoid multiple sub-intents, we used clickable buttons for the user to choose the options and read further about the topic.

We used the following process to build the bot.

1. ***Identify Intents***: The intents are the labels or class that are used by the machine learning model to map questions to the answer. To identify the main intents, it is important to find the concepts that students may wish to learn. We considered the entities from our ontology framework to provide the key concepts. We used the knowledge graphs to view and analyze the conversation flow and created the intents from the list of entities. There were 30 most important main intents chosen from the entity list. Additionally, there is a "greetings" intent for the bot to give a welcome message and "other" intent to gracefully continue the conversation for out-of-context questions.
2. ***Question-Answer Data Preparation***: As a next step, the questions were prepared based on these intents. We collected the frequently asked questions in emails from the instructor and TA of the course, discussion forums and quiz questions from previous years on the course portal. The questions were paraphrased for model training and were mapped to corresponding intents using a JSON file.
3. ***Intent Classification Model***: In order to classify the intents from the natural language questions, we used the SVM model as it is an extremely successful NLP technique for text classification, especially when the training dataset is not large enough [54]. SVM has been one of the most widely used models for multiple applications for over a decade [55].

   To develop the model, we first created features using the TF-IDF (term frequency-inverse document frequency) vectorization method. It calculates the weighted term frequency score for each word in the corpus relative to the document in the form of a vector. Thus, it gives the measure of how often that word is seen in the document by computing the overall document weightage of a word. To extract a bag of words, we used unigrams. We initially started with bigrams, but upon performing the "chi2test" and after analyzing the correlated terms, we found that the model prediction was better in the case of unigrams. This is because of the nature of questions in our dataset. Most questions were straightforward or how-to questions such as "*How to install*

Metasploit?" The linear-SVM model was trained for intent classification. The dataset contained the question ID and the questions as input and intents as labels. SVM returns the highest probability intent as the output. The predicted intent was sent to the json file where the intent is mapped to the corresponding answer, which is shown in the Chatbot messaging window in the UI. The current model reported a prediction accuracy of 92%. We plan to retrain the model on more questions that are being collected in the logs to improve the accuracy.

In this work, we considered using only the single model based on the main intents as labels. For sub-intents, we populated the choices in the response window as clickable buttons that students can click to know more about the topic. For example, as shown in Figure 7, if the user asks a question such as "Tell me about firewall?", the model then maps the question to the main intent, "*firewall*", and the UI provides the response that contains the definition of "firewall" with two buttons "IPTables" and "Run Firewalls". The students can click on buttons to know more about configuring IP tables rules of linux firewall or run the script for setting the firewall. This approach allowed us to have minimal ambiguity for the model and also to retained a light-weight application by using a single model in contrast to running multiple models for sub-intents at real time. The model allows students to paraphrase and ask the question in their natural language style. They can also go back and forth to the messaging window to view responses.

The implementation code of all the above components in this section is provided in our github repo [56]. Figure 8 shows the complete pipeline of our framework.

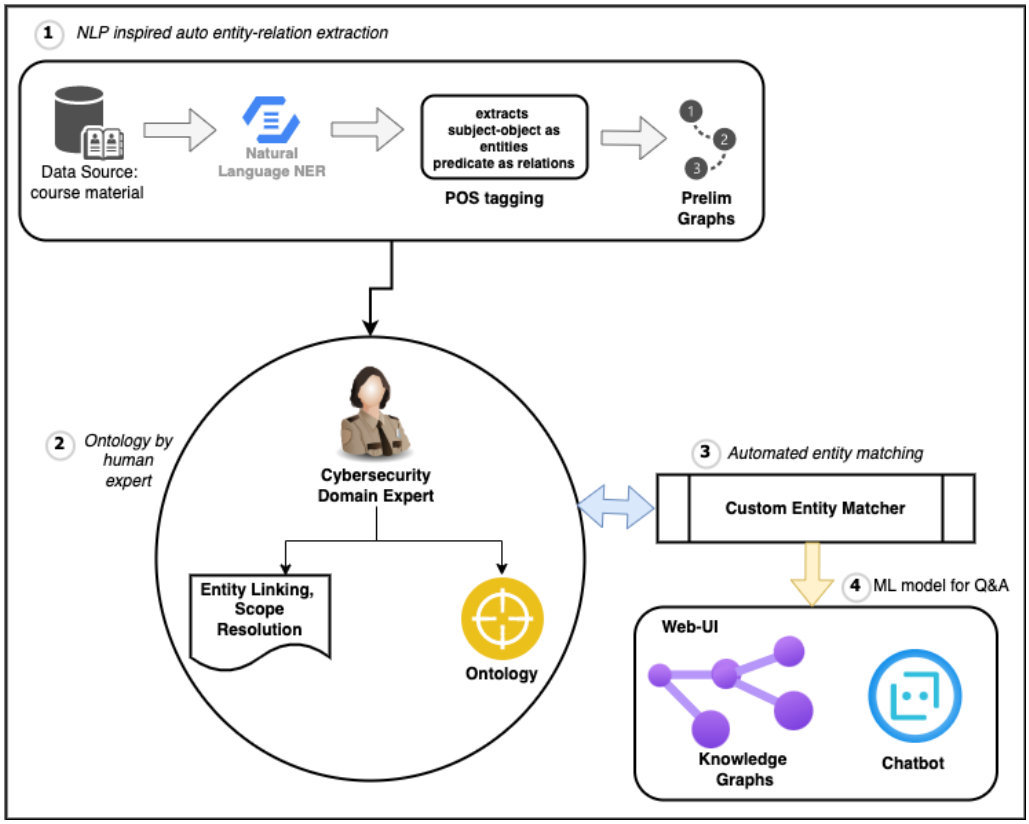


Figure 8. The pipeline of our framework as illustrated in Section 3. Step (1) shows the knowledge acquisition and automated entity extraction using NER, as given in Section 3.1.2. Step (2) shows the knowledge integration and ontology development by the human domain expert, as explained in Sections 3.1.3 and 3.1.4. The key entities and relations identified in (2) are given to custom entity matcher in Step (3), as explained in Section 3.1.5. The resulting triples are used in Step (4) to view the knowledge graphs and build other downstream tasks such as question answering, as shown in Section 3.3.

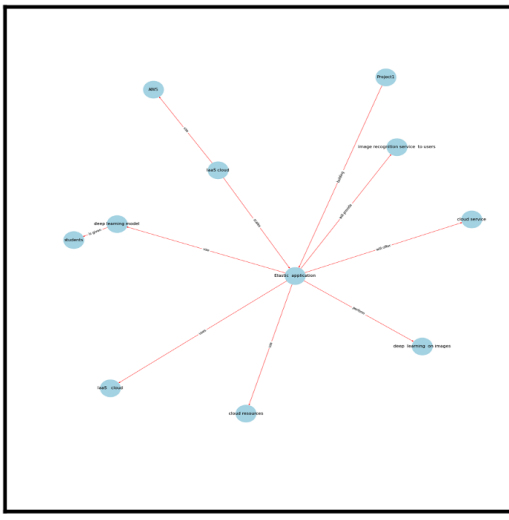
4. Evaluation

In this section, we first show that the proposed pipeline to construct the ontology and knowledge graphs can also be used in a different domain other than cybersecurity. In the second part, we gave the analysis of the impact of using knowledge graphs from student surveys and interview results.

4.1. Evaluation of Proposed Method

To evaluate our method, we tested the proposed pipeline on another domain called cloud computing. The course was taken by a different section of graduate students. For testing purposes, only a small subset of the course material was considered. We took the lab instruction manuals from the first two projects. The first project was based on building an elastic app to provide image recognition service using AWS and the second project was about implementing a smart classroom assistant for educators using cloud programming. The project documents were two-page each in the form of unstructured texts. On applying the sentence segmentation, we could obtain around 70 useful sentences. There were 62 raw entities and 44 relations extracted using the NER parts of speech tagging method out of which many were redundant. The temporary triples were used to generate the preliminary graphs from extracted entities and relations.

Figure 9a shows the preliminary graph generated from raw entities and relations for building the cloud app. The second author is the domain expert and instructor for cloud computing. The domain knowledge was used to remove the noise and redundant entities. The key entities and relations were identified from the preliminary visual graphs. The custom entity matcher was used on the third project, which is about building a hybrid cloud application. The sentences with key entities were easily identified using the entity matcher. Figure 9b shows the sample of key entities and their types for cloud computing domain. After the schema definition is established, the corresponding constraints as per domain and schema validation rules can then be created such that every DDL operation follows the schema. Thus, it shows that the proposed pipeline can be effectively used in identifying the key entities and ontology construction from the unstructured text for any new domain, especially in the absence of a well-defined ontology or structured triple dataset.



entityType	entityName
application	Elastic application, cloud app
service models	IaaS, PaaS, SaaS
process	Auto Scaling, Elastic Load Balancing, Scaling, multi-threading
services	AWS, Amazon S3, Google App Engine, Amazon EC2, Heroku, Workday, Google App Engine, Simple Notification Service (SNS), Simple Queue Service (SQS)
Infrastructure	cloud, network, servers, operating system, storage, client devices, Amazon Machine Images, virtual computing environment
Deployment	on-premise, public, private cloud, hybrid cloud
tasks	image recognition, smart classroom, deep learning, project1
user	Student, users, cloud admin

(a) (b)

Figure 9. Cloud computing entity extraction (**a**) The first figure represents the preliminary graph generated by extracting raw entities and relations from unstructured text (**b**) The second figure shows the key entities identified by the domain expert using the raw entities and visual graphs.

4.2. Impact Analysis of Knowledge Graphs

To assess the perception of students on using the knowledge graphs to learn cybersecurity concepts and solve the problem-based learning challenges, we conducted surveys

and interviews with students. We will further illustrate the students' perceptions and experiences using these methods and also provide a summary analysis of responses from the students.

The knowledge graphs were developed for the course on 'Advanced Network Security', which was taught to graduate students majoring in computer science and cybersecurity fields.

We adapted the Self-Efficacy and Metacognition Learning Inventory-Science (SEMLI-S) survey, which was proposed by Thomas et al. [57]. The SEMLI-S survey is designed to investigate the perception of science students' metacognition, self-efficacy and learning processes. The SEMLI-S method uses five dimensions of students' perception, viz., Constructivist Connectivity (CC), Monitoring, Evaluation and Planning (MEP), Science Learning Self-efficacy (SE), Learning Risk Awareness (AW) and Control of Concentration (CO). These five dimensions are designed to measure the self-regulated constructive learning process of students [57], which is one of the core pillars of problem-based learning [58]. This survey methodology was deemed fit for our study as knowledge graphs are being used in problem-based learning tasks in cybersecurity labs.

Some of the original questions were rephrased in the context of knowledge graphs; for example, an original survey question in the method "I seek to connect the information in science class with what I already know" was revised to "Knowledge graph helps me connect what I learn in this course with what I already know". However, some original questions (e.g., "I am confident I can do a good job on the assignments and tests in this science class") were removed as they were not required in the context of problem-solving with knowledge graphs.

A total of 18 questions were included in the survey. The same survey was given twice to students after each cybersecurity lab project. The purpose was to assess how the students perceived the usefulness of knowledge graphs in completing the lab tasks and also to find whether their perception had changed throughout the two problem-based learning sessions. Participation in the survey was not made mandatory. From the class of 27, a total of 21 students participated in the survey when it was conducted the first time and 15 students responded the second time. Overall, about 76.2% of the students in the first survey perceived the KG as either somewhat, or very useful for their problem-based learning tasks. The percentage of these categories of respondents slightly increased in the second survey to 86.7%.

The surveys were followed by semi-structured interviews to obtain an in-depth understanding of the students' perceptions using knowledge graphs to solve lab challenges. The students were asked during the survey if they would be willing to participate in the follow-up interview. A total of 18 students agreed to participate in the individual interviews that were conducted after the students had completed the project tasks. The interview meetings were held via ZOOM meeting and lasted about 20–30 min each. A general interview protocol was set up by the third author and reviewed and revised by other authors. The primary objective of the interviews was to explore three aspects of students' experiences of using the knowledge graphs: (a) perceived usefulness, (b) ways and strategies for using the KG, and (c) suggestions for any improvement of KG. The examples of interview questions include the following: "How did you like the knowledge graph? Was it useful?", "Which features of the knowledge graph were helpful or unhelpful?", "When and how did you use the knowledge graph?", and "Based on your experiences, do you have any suggestions for improving the knowledge graph?" Some flexibility in sequencing the questions and specifying answers was made to capture the characteristics of individual experiences. Individual adaptations for specific questions were also made according to the survey data of students; for example, a question such as "Why KG was not much helpful for monitoring the learning progress (i.e., MEP)" was asked only from students who rated lower scores in the survey. All interview responses were transcribed for analyses. The transcribed data were then open-coded using a constant comparative method. During this process, a codebook was generated, which reflects positive and nega-

tive experiences, learning strategies and recommendations/suggestions. Using the codes, in-depth examinations were again conducted to categorize the students' experiences into several common emerging themes.

Results

Upon analyses of surveys and interview responses, there were three salient themes of students' perceived usefulness of the knowledge graph and chatbot. Table 2 shows the identified features and evidence from the surveys and interviews.

Table 2. Evaluation themes and evidences from survey and interviews.

Theme 1—Students find the knowledge graph as an informative tool
Summary—Students found knowledge graphs useful as an informative tool to learn the core concepts. It also helped them understand the problem structure and gave a flow map of the tasks assigned to them. **Evidence from Open-ended Questions**—"The knowledge graph was the basis for me understanding the overarching concepts and connection between parts." **Evidence from Interview**—"I found to be extremely helpful, I definitely was able to use the knowledge graph to see like okay here are the concepts that I need to know." "Yeah, I would say, I liked it basically because the knowledge graph gives you a skeletal structure of how the flow should be (look like)."
Theme 2—Students use the knowledge graph as a visual reference
Summary—They used the knowledge graphs for monitoring and visually cross checking their progress on project tasks. **Evidence from Open-ended Questions**—"It (Knowledge graph) gave the exact visual view of the project." **Evidence from Interview**—"I pull knowledge graph up, real quick, so I can actually have a visual reference because I did use it throughout the labs." "(Knowledge graph) is used, for like, a solid ground. Just kind of, confirming that I am going the right direction."
Theme 3: Chatbot and the knowledge graph are an easy access to just-in-time information
Summary—The chatbot and KG were useful as an on-demand learning tool for immediate feedback and answering the how-to questions for implementing a task. **Evidence from Interview**—"(I) immediately go back and see what is that, how do I use this, how do I implement this." "Go to the knowledge graph and it would quickly take me to the Wiki link with a few clicks that was, that was pretty convenient."

4.3. Discussion

In this research work, our focus was to propose a knowledge graph construction pipeline for a domain that lacked a well-defined ontology or standard dataset. Using our method, we showcased the process of identifying the key entities, attributes and relations to build the ontology framework for the cybersecurity education domain. To ensure the consistency of incoming data in the knowledge graph, we used schema-aware property graphs by applying schema validation. We also show that the proposed method can be effectively used for constructing knowledge graphs for other domains such as cloud computing.

Another objective of this paper was to show that knowledge graphs can be an effective and handy tool for students in promoting problem-based learning. The results from the surveys and interviews with the students were encouraging and showed that knowledge graphs helped them set up and install the system and gave an idea of how to build the projects. The students mentioned that it gave them a kind of visual map of how different components interact with each other and a skeletal structure of the flow. The knowledge graph was helpful because they thought it gave them a hint about how to get started and information on what actions they were supposed to perform on a particular task. The students mentioned that, by using knowledge graphs, they could confirm that they were

going in the right direction. Some of them said they used it throughout the lab for all projects. The bot gave immediate answers to the how-to questions.

Out of the 18 students who took at least one of the surveys and interviews, almost 85% of students found the knowledge graphs useful in solving problem-based tasks in learning cybersecurity, whereas 15% found the graphical view confusing and crowded. Some of them thought they did not need such a tool for simpler tasks in the projects. However, they used it only when the project was complex. These were some of the good takeaways ,and we plan to apply this feedback to further improve our work.

Some of the other works such as Yang Chi et al. [18] used knowledge graphs in teaching scientific papers. Their survey results show learning improvement in students. Gong et al. [59] used knowledge graphs to teach Python coding online. Their results also showed that knowledge graphs were effective for students with no foundation in Python. Our survey results from the SEMLI-S survey methods were in agreement with other studies and showed that knowledge graphs were useful in promoting education in cybersecurity. In future work, we plan to enrich the entities in order to make it more comprehensive and publish the triple dataset for cybersecurity education. We also plan to build embedding models for semantic reasoning to provide learning recommendations for students.

5. Conclusions

We show a methodology to construct knowledge graphs from unstructured course material in the absence of any standard ontology or dataset. We used a bottom-up approach to first extract the raw entities and relations from text using natural language processing. We then used human expert domain knowledge to refine the attributes for entities and relations and proposed a schema to create triples related to cybersecurity education domain. The visual graphs were constructed for project tasks and an interactive chatbot was built to answer the student queries based on the key entities. We built a web-based UI to provide these tools to graduate students. We tested our proposed pipeline on cloud computing domain and showed that the approach can be used to construct the ontology framework from unstructured text for any domain. To analyze the impact of using knowledge graphs as a learning aid in problem-solving tasks, we conducted surveys and interviews with students to assess their perception on using these tools. The survey results show that students found knowledge graphs to be informative in giving an overall flow map, and they used it often as a visual reference to monitor their progress on the project.

In the future, we plan to extend this work to build downstream applications such as curriculum design and learning recommendation tools. There is huge potential for further research and applications in this space, which can benefit not only cybersecurity students and professionals but also other domains. The research methodology proposed in this work can be very well adapted by other domain researchers for constructing knowledge graphs.

Author Contributions: Conceptualization, G.A., Y.D. and J.P.; methodology, G.A.; software, G.A.; validation, G.A. and Y.D.; formal analysis, G.A.; investigation, G.A., Y.D. and J.P.; survey and interviews, J.P., Y.-C.C. and Y.D.; resources, Y.D., J.P. and Y.-C.C.; data curation, G.A.; writing—original draft preparation, G.A.; writing—review and editing, G.A., Y.D. and J.P.; visualization, G.A.; supervision, Y.-C.C. and H.L.; project administration, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This material is based upon work supported by the National Science Foundation under Grant No. 2114789. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Data Availability Statement: Not applicable.

Acknowledgments: We would like to acknowledge Dijiang Huang for his vision, ideas and guidance.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

KG	Knowledge graphs;
KGQA	Knowledge graph question answering systems;
SVM	Support vector machine;
IDS	Intrusion detection system;
NER	Named entity recognition;
NLP	Natural language processing;
UI	User interface.

References

1. Singhal, A. Introducing the Knowledge Graph: Things, Not Strings. May 2012. Official Blog, of Google. 2012. Available online: http://googleblog.blogspot.ie/2012/05/introducing-knowledgegraph-things-not.html (accessed on 10 September 2022).
2. Sowa, J.F. Semantic networks. *Encycl. Cogn. Sci.* **2012**. Available online: http://www.jfsowa.com/pubs/semnet.pdf (accessed on 10 September 2022).
3. Hogan, A.; Blomqvist, E.; Cochez, M.; d'Amato, C.; Melo, G.D.; Gutierrez, C.; Kirrane, S.; Gayo, J.E.L.; Navigli, R.; Neumaier, S.; et al. Knowledge graphs. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–37. [CrossRef]
4. Wu, X.; Wu, J.; Fu, X.; Li, J.; Zhou, P.; Jiang, X. Automatic knowledge graph construction: A report on the 2019 icdm/icbk contest. In Proceedings of the 2019 IEEE International Conference on Data Mining (ICDM), Beijing, China, 8–11 November 2019; pp. 1540–1545.
5. Kang, S.J.; Kang, I.S. Generalization of Ontology Instances Based on WordNet and Google. *J. Korean Inst. Intell. Syst.* **2009**, *19*, 363–370.
6. Gould, N.; Mackaness, W. From taxonomies to ontologies: Formalizing generalization knowledge for on-demand mapping. *Cartogr. Geogr. Inf. Sci.* **2016**, *43*, 208–222. [CrossRef]
7. Lin, J.; Zhao, Y.; Huang, W.; Liu, C.; Pu, H. Domain knowledge graph-based research progress of knowledge representation. *Neural Comput. Appl.* **2021**, *33*, 681–690. [CrossRef]
8. Jung, Y.; Ryu, J.; Kim, K.M.; Myaeng, S.H. Automatic construction of a large-scale situation ontology by mining how-to instructions from the web. *Web Semant. Sci. Serv. Agents World Wide Web* **2010**, *8*, 110–124. [CrossRef]
9. Shin, J.; Wu, S.; Wang, F.; De Sa, C.; Zhang, C.; Ré, C. Incremental knowledge base construction using deepdive. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*; NIH Public Access: Bethesda, MD, USA, 2015; Volume 8, p. 1310.
10. Guo, L.; Yan, F.; Li, T.; Yang, T.; Lu, Y. An automatic method for constructing machining process knowledge base from knowledge graph. *Robot. Comput.-Integr. Manuf.* **2022**, *73*, 102222. [CrossRef]
11. Issa, S.; Adekunle, O.; Hamdi, F.; Cherfi, S.S.S.; Dumontier, M.; Zaveri, A. Knowledge graph completeness: A systematic literature review. *IEEE Access* **2021**, *9*, 31322–31339. [CrossRef]
12. Kejriwal, M. *Domain-Specific Knowledge Graph Construction*; Springer: Berlin/Heidelberg, Germany, 2019.
13. Chaudhri, V.; Baru, C.; Chittar, N.; Dong, X.; Genesereth, M.; Hendler, J.; Kalyanpur, A.; Lenat, D.; Sequeda, J.; Vrandečić, D.; et al. Knowledge Graphs: Introduction, History and, Perspectives. *AI Mag.* **2022**, *43*, 17–29.
14. Abu-Salih, B. Domain-specific knowledge graphs: A survey. *J. Netw. Comput. Appl.* **2021**, *185*, 103076. [CrossRef]
15. Pan, J.Z.; Vetere, G.; Gomez-Perez, J.M.; Wu, H. *Exploiting Linked Data and Knowledge Graphs in Large Organisations*; Springer: Berlin/Heidelberg, Germany, 2017.
16. Qin, Y.; Cao, H.; Xue, L. Research and Application of Knowledge Graph in Teaching: Take the database course as an example. In *Proceedings of the Journal of Physics: Conference Series*; IOP Publishing: Bristol, UK, 2020; Volume 1607, p. 012127.
17. Sun, K.; Liu, Y.; Guo, Z.; Wang, C. Visualization for knowledge graph based on education data. *Int. J. Softw. Inform.* **2016**, *10*, 1–13.
18. Chi, Y.; Qin, Y.; Song, R.; Xu, H. Knowledge graph in smart education: A case study of entrepreneurship scientific publication management. *Sustainability* **2018**, *10*, 995. [CrossRef]
19. Deng, L.; Liu, Y. *Deep Learning in Natural Language Processing*; Springer: Berlin/Heidelberg, Germany, 2018.
20. Kertkeidkachorn, N.; Ichise, R. T2kg: An end-to-end system for creating knowledge graph from unstructured text. In Proceedings of the Workshops at the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
21. Zhao, M.; Wang, H.; Guo, J.; Liu, D.; Xie, C.; Liu, Q.; Cheng, Z. Construction of an industrial knowledge graph for unstructured chinese text learning. *Appl. Sci.* **2019**, *9*, 2720. [CrossRef]
22. Sant'Anna, D.T.; Caus, R.O.; dos Santos Ramos, L.; Hochgreb, V.; dos Reis, J.C. Generating Knowledge Graphs from Unstructured Texts: Experiences in the E-commerce Field for Question Answering. In Proceedings of the ASLD@ ISWC, Athens, Greece, 1–6 November 2020; pp. 56–71.
23. Chen, Y.; Kuang, J.; Cheng, D.; Zheng, J.; Gao, M.; Zhou, A. AgriKG: An agricultural knowledge graph and its applications. In Proceedings of the International Conference on Database Systems for Advanced Applications, Chiang Mai, Thailand, 22–25 April 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 533–537.

24. Martinez-Rodriguez, J.L.; López-Arévalo, I.; Rios-Alvarado, A.B. Openie-based approach for knowledge graph construction from text. *Expert Syst. Appl.* **2018**, *113*, 339–355. [CrossRef]
25. Noy, N.; Gao, Y.; Jain, A.; Narayanan, A.; Patterson, A.; Taylor, J. Industry-scale Knowledge Graphs: Lessons and Challenges: Five diverse technology companies show how it's done. *Queue* **2019**, *17*, 48–75. [CrossRef]
26. Jain, N. Domain-specific knowledge graph construction for semantic analysis. In Proceedings of the European Semantic Web Conference, Crete, Greece, 31 May–4 June 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 250–260.
27. Shi, B.; Weninger, T. Open-world knowledge graph completion. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32.
28. Lenat, D.B. CYC: A large-scale investment in knowledge infrastructure. *Commun. ACM* **1995**, *38*, 33–38. [CrossRef]
29. Subasic, P.; Yin, H.; Lin, X. Building Knowledge Base through Deep Learning Relation Extraction and Wikidata. In Proceedings of the AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering, Palo Alto, CA, USA, 25–27 March 2019.
30. Wang, K.; Shen, Z.; Huang, C.; Wu, C.H.; Dong, Y.; Kanakia, A. Microsoft academic graph: When experts are not enough. *Quant. Sci. Stud.* **2020**, *1*, 396–413. [CrossRef]
31. Souag, A.; Salinesi, C.; Mazo, R.; Comyn-Wattiau, I. A security ontology for security requirements elicitation. In Proceedings of the International Symposium on Engineering Secure Software and Systems, Milan, Italy, 4–6 March 2015; Springer: Berlin/Heidelberg, Germany, 2015; pp. 157–177.
32. Mozzaquatro, B.A.; Agostinho, C.; Goncalves, D.; Martins, J.; Jardim-Goncalves, R. An ontology-based cybersecurity framework for the internet of things. *Sensors* **2018**, *18*, 3053. [CrossRef]
33. Jia, Y.; Qi, Y.; Shang, H.; Jiang, R.; Li, A. A practical approach to constructing a knowledge graph for cybersecurity. *Engineering* **2018**, *4*, 53–60. [CrossRef]
34. Bürger, J.; Strüber, D.; Gärtner, S.; Ruhroth, T.; Jürjens, J.; Schneider, K. A framework for semi-automated co-evolution of security knowledge and system models. *J. Syst. Softw.* **2018**, *139*, 142–160. [CrossRef]
35. Doynikova, E.; Fedorchenko, A.; Kotenko, I. Ontology of metrics for cyber security assessment. In Proceedings of the 14th International Conference on Availability, Reliability and Security, Canterbury, UK, 26–29 August 2019; pp. 1–8.
36. Alenezi, M.; Basit, H.A.; Khan, F.I.; Beg, M.A. A Comparison Study of Available Sofware Security Ontologies. In Proceedings of the Evaluation and Assessment in Software Engineering, Trondheim, Norway, 15–17 April 2020; pp. 499–504.
37. Rizun, M. Knowledge graph application in education: A literature review. *Acta Univ. Lodz. Folia Oeconomica* **2019**, *3*, 7–19. [CrossRef]
38. Chen, P.; Lu, Y.; Zheng, V.W.; Chen, X.; Yang, B. Knowedu: A system to construct knowledge graph for education. *IEEE Access* **2018**, *6*, 31553–31563. [CrossRef]
39. Chen, P.; Lu, Y.; Zheng, V.W.; Chen, X.; Li, X. An automatic knowledge graph construction system for K-12 education. In Proceedings of the Fifth Annual ACM Conference on Learning at Scale, London, UK, 26–28 June 2018; pp. 1–4.
40. Aliyu, I.; Kana, A.; Aliyu, S. Development of knowledge graph for university courses management. *Int. J. Educ. Manag. Eng.* **2020**, *10*, 1. [CrossRef]
41. Deng, Y.; Lu, D.; Huang, D.; Chung, C.J.; Lin, F. Knowledge graph based learning guidance for cybersecurity hands-on labs. In Proceedings of the ACM Conference on Global Computing Education, Chengdu, China, 17–19 May 2019; pp. 194–200.
42. Deng, Y.; Zeng, Z.; Huang, D. Neocyberkg: Enhancing cybersecurity laboratories with a machine learning-enabled knowledge graph. In Proceedings of the 26th ACM Conference on Innovation and Technology in Computer Science Education V. 1, Virtual Event, Germany, 26 June–1 July 2021; pp. 310–316.
43. Deng, Y.; Zeng, Z.; Jha, K.; Huang, D. Problem-Based Cybersecurity Lab with Knowledge Graph as Guidance. *J. Artif. Intell. Technol.* **2022**, *2*, 55–61. [CrossRef]
44. Brank, J.; Grobelnik, M.; Mladenic, D. A survey of ontology evaluation techniques. In Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005), Ljubljana, Slovenia, 17 October 2005; pp. 166–170.
45. Bonifati, A.; Furniss, P.; Green, A.; Harmer, R.; Oshurko, E.; Voigt, H. Schema validation and evolution for graph databases. In Proceedings of the International Conference on Conceptual Modeling, Salvador, Brazil, 4–7 November 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 448–456.
46. ReGraph Documentation. Available online: http://dev.executableknowledge.org/ReGraph/ (accessed on 25 October 2022).
47. Corradini, A.; Heindel, T.; Hermann, F.; König, B. Sesqui-pushout rewriting. In Proceedings of the International Conference on Graph Transformation, Rio Grande do Norte, Brazil, 17–23 September 2006; Springer: Berlin/Heidelberg, Germany, 2006; pp. 30–45.
48. Punnoose, R.; Crainiceanu, A.; Rapp, D. Rya: A scalable RDF triple store for the clouds. In Proceedings of the 1st International Workshop on Cloud Intelligence, Istanbul, Turkey, 31 August 2012; pp. 1–8.
49. Miller, J.J. Graph database applications and concepts with Neo4j. In Proceedings of the Southern Association for Information Systems Conference, Atlanta, GA, USA, 23–24 March 2013; Volume 2324.
50. Francis, N.; Green, A.; Guagliardo, P.; Libkin, L.; Lindaaker, T.; Marsault, V.; Plantikow, S.; Rydberg, M.; Selmer, P.; Taylor, A. Cypher: An evolving query language for property graphs. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 1433–1445.

51. Angles, R.; Arenas, M.; Barceló, P.; Hogan, A.; Reutter, J.; Vrgoč, D. Foundations of modern query languages for graph databases. *ACM Comput. Surv. (CSUR)* **2017**, *50*, 1–40. [CrossRef]
52. Angles, R.; Arenas, M.; Barceló, P.; Boncz, P.; Fletcher, G.; Gutierrez, C.; Lindaaker, T.; Paradies, M.; Plantikow, S.; Sequeda, J.; et al. G-CORE: A core for future graph query languages. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 1421–1432.
53. van Rest, O.; Hong, S.; Kim, J.; Meng, X.; Chafi, H. PGQL: A property graph query language. In Proceedings of the Fourth International Workshop on Graph Data Management Experiences and Systems, Redwood Shores, CA, USA, 24 June 2016; pp. 1–6.
54. Matykiewicz, P.; Pestian, J. Effect of small sample size on text categorization with support vector machines. In Proceedings of the BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing, Montreal, QC, Canada, 8 June 2012; pp. 193–201.
55. Luo, X. Efficient english text classification using selected machine learning techniques. *Alex. Eng. J.* **2021**, *60*, 3401–3409. [CrossRef]
56. Agrawal, G. KG for Cybersecurity Education. 2022. Available online: https://github.com/garima0106/KG-Cybersec.git (accessed on 3 November 2022).
57. Thomas, G.; Anderson, D.; Nashon, S. Development of an instrument designed to investigate elements of science students' metacognition, self-efficacy and learning processes: The SEMLI-S. *Int. J. Sci. Educ.* **2008**, *30*, 1701–1724. [CrossRef]
58. Loyens, S.M.; Magda, J.; Rikers, R.M. Self-directed learning in problem-based learning and its relationships with self-regulated learning. *Educ. Psychol. Rev.* **2008**, *20*, 411–427. [CrossRef]
59. Gong, Z.; Yu, X.; Fu, W.; Che, X.; Mao, Q.; Zheng, X. The Construction of Knowledge Graph for Personalized Online Teaching. In Proceedings of the International Conference on Data Mining and Big Data, Guangzhou, China, 20–22 October 2021; Springer: Berlin/Heidelberg, Germany, 2021; pp. 98–107.