IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 5, 2021, accepted October 29, 2021, date of publication November 30, 2021,
date of current version December 14, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3131627

Automating Computer Science Ontology Extension With Classification Techniques

NATASHA C. SANTOSA[1], JUN MIYAZAKI[1], (Member, IEEE), AND HYOIL HAN[2]
[1]Department of Computer Science, School of Computing, Tokyo Institute of Technology, Tokyo 152-8550, Japan
[2]School of Information Technology, Illinois State University, Normal, IL 61790, USA

Corresponding author: Jun Miyazaki (miyazaki@c.titech.ac.jp)

This work was supported in part by Japan Society for the Promotion of Science (JSPS) KAKENHI Grant Numbers JP18H03242, JP18H03342, and JP19H01138.

ABSTRACT In information technology, an ontology is a knowledge structure consisting of terminologies (topics), their definitions, and relational information within one or multiple domains. This semantically represented information can be used for downstream tasks, such as document classification and recommendation systems. However, as big data prevails, manually extending existing ontologies with up-to-date information becomes challenging due to the tedious and time-consuming process and the expensive cost of expert manual labor. To alleviate this problem, this paper aims to achieve a fully automatic ontology extension. We propose a novel ''Direct'' approach for extending an existing Computer Science Ontology (CSO). This approach consists of two steps: initially extending the CSO with new topics and using this extended graph to obtain the new topic's node embeddings as inputs for training classifiers. However, this initial extension still contains numerous noisy links; therefore, the classifier simultaneously acts as a noisy-link filter and a link predictor. We experiment with various traditional machine learning and recent deep learning models and then compare them under our Direct approach framework. We propose two evaluation procedures to decide the best-performing model: a novel Wikipedia-based $F1_w$ score and a total number of resulting links. Further meta-evaluation employing four experts confirmed the reliability of our proposed approach and evaluation procedures. We found that the Direct approach's *Gaussian Naive Bayes* model produces the most valid and reliable links; therefore, we use it to further extend the CSO with hundreds of new CS topics and links.

INDEX TERMS Classification methods, concept classification, deep learning, link prediction, ontology extension.

I. INTRODUCTION

Organizing information is becoming more crucial due to the rapid growth of information. Proper information organization allows for easier access to certain information from a vast pool of information. Ontology is a popular data structure used to store information efficiently; it is used to store terminologies (topics within a domain) and the relationships among them. Our long term goal is to integrate ontologies into recommendation systems; which would be beneficial for enhancing semantic reconciliation among data sources and improving recommendation explainability. However, a major problem here is the maintenance of the ontology itself, that is, we need to keep ontologies up-to-date while minimizing human intervention. Maintenance means updating an existing ontology with new information. This includes adding new topics and updating the existing relationships. Several methodologies exist for the ontology maintenance, which often include corpus annotation, training information extraction engine, and validation by human experts.

There has been a lot of studies on ontology construction in the last two decades, and diverse methods to construct ontology have been proposed. In the early 2000s, various studies tried to produce multiple semiautomatic tools for constructing an ontology from raw texts, for example, news articles, scientific publications, and product reviews [1]–[3]. However, the semiautomatic approaches for ontology construction still rely heavily on expert human labor, and only decreases the effort of ontology construction by approximately $\leq 50\%$ compared to constructing it manually (cf. Section II-D). This is undesirable because methods that require human interventions at any of its sub-process are non-scalable for future changes [4].

The associate editor coordinating the review of this manuscript and approving it for publication was Sotirios Goudos.

Rather than building an ontology from scratch, utilizing an already existing one is preferable as it saves time and effort. In this paper, we specifically experiment with the **Computer Science Ontology**[1] (CSO) [5] that stores information about Computer Science research topics, for example, *machine learning* and *human-computer interaction*. To the best of our knowledge, this is the most up-to-date and exhaustive Computer Science topic ontology by far. However, the CSO project still relies on semi-manual maintenance. Particularly, it relies on a human-expert to filter-out and cross-check automatic machine suggestions (cf. Section III).

We explore and evaluate various approaches to find the most appropriate one for realizing a completely automated ontology maintenance, specifically the ontology extension process. The main contributions of this paper are threefold:

1) We propose a novel "Direct" automatic extension approach. Our approach differs from the existing "Indirect" approach (cf. Section II) in that we *directly involve* new topics when training classifier models to predict the existence of links between topics (while the existing approach did not).
2) We compare the performance of the two approaches via a novel Wikipedia-based evaluation procedure as well as the total number of valid links predicted by the classifiers. In the past, expert human cross-checking was required when updating an existing ontology. However, human labor is time consuming and prone to inconsistency. Wikipedia is a very comprehensive body of knowledge, and it is updated frequently due to its open-collaboration system. Furthermore, there have also been multiple works that relied on Wikipedia as their main source of information (cf. Section II-G). Our proposed evaluation procedure based on Wikipedia enables cheaper and faster assessment of the ontology extension task. We additionally conducted a meta-evaluation by employing four experts and showed that the experts tend to highly agree with the links predicted by the top three best performing classifiers. This shows that by using only the Wikipedia-based evaluation procedure and the number of total valid links, we can arrive at a similar conclusion to employing expert judges.
3) As the result of our effort, we are able to extended the CSO with approximately 1,716 new links and 1,140 new computer science topics (approx. 8.14% increase of new topics).

A. RESEARCH QUESTIONS

This paper aims to answer the following research questions:

1) Is a complete automated ontology extension approach feasible?
2) Which approach, direct or indirect, is better at recognising new topics and properly linking them into the CSO ontology?

[1]https://cso.kmi.open.ac.uk/downloads

B. PAPER STRUCTURE

The remainder of this paper is organized as follows. Section II describes a literature review that includes the general definition of ontology and several related works within the field of ontology extension. In Section III, we describe the ontology data used and how this ontology is obtained. The methodology of each approach proposed in this study is explained in Sections IV and V. Section VI describes the details of our experiments and discusses the models' performance in-depth. Finally, the conclusions of this study and further study plans are explained in Section VII.

II. RELATED WORKS

A. ONTOLOGY IN COMPUTER SCIENCE

Man [6] defined ontology in Computer Science as a formal representation of knowledge consisting of a set of topics under a domain and the relationships between these topics. In other words, an ontology is a description of the body of knowledge for a domain. This method of knowledge representation has been used in several fields including artificial intelligence, semantic web, systems engineering, and many more. In this paper, ontology is used as a knowledge representation of scientific paper topics (under the Computer Science domain) and their relations.

In the case of preexisting ontologies, **ontology maintenance** such as extending ontology with updated concepts should be done as well. For the simplicity of referring to the idea of maintaining an existing ontology with new up-to-date information, we refer to it as **ontology extension** in the rest of this paper.

B. TAXONOMY EXTENSION

Before proceeding to the next section, it is important to mention that there have been numerous studies on methods for taxonomy extension. Taxonomy extension, like ontology extension, is the action of enriching an already existing taxonomy with new information. For example, HiExpan [4] allowed users to expand a taxonomy either horizontally or vertically by referring to a given seed taxonomy. TaxoGen [7], a similar idea to HiExpan, used an unsupervised approach to automatically construct a topic taxonomy from raw texts given a seed. Even though TaxoGen [7] focuses more on taxonomy construction from scratch, we believe the idea can be further expanded to an extension task when a seed taxonomy is introduced to the process, due to the similarities in the processes involved.

Despite the potential of applying an automatic taxonomy extension method for ontology extension, existing approaches are nevertheless not suitable for our purpose for two reasons. First, the CSO is a combination of tree and ordinary graph structures—hierarchy presents (supertopic–subtopic relationship), but at the same time, it does not strictly adhere to the hierarchical rules of a tree or taxonomy. For example, a certain node in the CSO may have multiple roles: as sibling and grandparent of a particular node, at the same

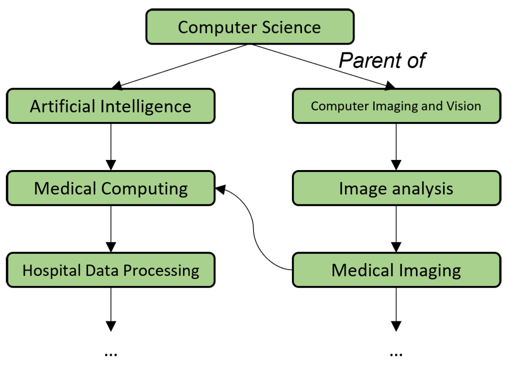

FIGURE 1. Unique structure of the computer science ontology.

time. Figure 1 shows a relationship snippet in the CSO. Here, the topic of "Medical Imaging" is at the same level as "Hospital Data Processing". However, at the same time, it is the parent of "Medical Computing" and, therefore, it is the grandparent of "Hospital Data Processing". Relationships within the CSO are far more complicated and should be principally treated as a graph. Therefore, taxonomy extension methods cannot be applied easily to the CSO. Second, the aforementioned taxonomy extension methods also relied on imperfect key term extractors, which affected the performance of the systems. This will be explained in more detail in the next section.

C. AUTOMATIC TERM EXTRACTION

An automatic term extractor aims to automatically extract important terminologies from a given corpus of raw texts [8]. There have been a lot of studies on this topic, and some of the produced automatic term extraction algorithms and tools include SemCluster [8], AutoPhrase [9], [10], YAKE [11]–[13], and RAKE [14]. The use of terms extractor algorithms or tools is very popular when extending taxonomy or ontology. For example, HiExpan [4] used AutoPhrase [9] to extract important terminologies from raw texts.

However, resulting outputs from automatic term extractors are often noisy, for example, the extractor may only recognize "Human Computer" while the gold standard is "Human Computer Interaction". This often has to do with the imperfection in named entity recognition. Another problem occurs when lemmatization is performed on raw texts, resulting in unnatural technical terms; for example, "Machine Learn" from a supposed gold term "Machine Learning".

In this paper, we do not rely on the automatic term extractor when training our system. However, we do use an existing term extractor during the evaluation process. Details of our approach are further explained in Section IV.

D. SEMIAUTOMATIC ONTOLOGY EXTENSION

Studies on ontology extension have mostly focused on the semiautomatic approach. This approach typically consists of two main steps. First, the model automatically extracts domain-specific terminologies from an information source (e.g., text documents) while providing the predictions or probabilities of the their supposed location within the ontology. Second, a human intervention is performed for checking, correcting, and confirming the automatically generated results from the earlier step.

Ayadi *et al.* [15] proposed a semiautomatic ontology extension approach by implementing a deep-learning-based extractor to recognize terminologies alongside their relations and attribute instances. Then, experts were relied on to completely check the outputs of their deep learning extraction model. On the other hand, Liu *et al.* [16] proposed a semiautomatic ontology extension and refinement. In their work, they used the WordNet lexical dictionary to create a semantic network from a given seed ontology. This semantic network was then processed by spreading activation, deciding whether a concept should be included in the seed ontology or not. Novalija and Mladenic [17] employed a combination of text mining and a user-oriented approach to semiautomatically extend a cross-domain Cyc ontology.[2] A follow up study in [18] extended their previous method by considering ontology content and structure.

These studies relied on human intervention in the process, specifically for cross-checking the system's outputs. However, it is difficult to find an expert who has comprehensive and in-depth knowledge of all concepts in one particular domain. Therefore, some (but limited) experts were employed in practice. In contrast, our proposed method which relies on Wikipedia (as ground truth) during the evaluation procedure, is a cheap alternative to employing a large number of experts or crowd workers.

E. RECENT ATTEMPT AT AN AUTOMATIC ONTOLOGY EXTENSION

Althubaiti *et al.* [19] attempted to realize a fully automated ontology extension by employing an artificial-neural-network-based word classifier. To the best of our knowledge, Althubaiti *et al.* [19] is the only one thus far that have attempted to fully automate the ontology extension task. Given a new disease terminology, their model linked it to its appropriate parent topic by utilizing their vector representations. Althubaiti *et al.* did not use the complete disease ontology terminologies. Instead, they only selected a few topics as potential parents (labels in classification) for a given new topic. They separated their classification types into four sets: disease (number of classes $= 2$), infectious disease (number of classes $= 5$), anatomical disease (number of classes $= 13$), and infectious $+$ anatomical disease (number of classes $= 17$).

Nevertheless, the work by Althubaiti *et al.* inspired one of the approaches we use in this paper as well as in our previous paper [20] where we observed the performance of flat and hierarchical classifiers when using Althubaiti *et al.*'s framework. Based on our previous experiment in [20], we decide to

[2]Cyc: https://cyc.com/archives/glossary/ontology/

only report the flat classifier models in this paper due to their better performance scores. In comparison to our previous work, here, we introduce a new approach called the "Direct" approach. We then empirically evaluate our new approach in comparison to Althubaiti *et al.*'s (referred to as the "Indirect" approach in the rest of this paper). Section IV explains our implementation of the Indirect approach.

F. LINK PREDICTION TASK

Given two terminologies (nodes in a graph): a new topic and a potential parent, the link prediction task aims to judge whether a link exists between them, i.e., a binary classification problem. However, this formulation leads to extremely sparse data as most of the topic pairs are not linked. The link prediction task can be alternately formulated as a multilabel classification task (which we adopt in this paper), as a new topic can have multiple parents. Given a new topic, a classifier model judges which existing node(s) (i.e., classification labels) we should link the new topic to.

The majority of link prediction algorithms rely on the use of topic embedding and similarity measures (e.g., cosine similarity) for concluding the final prediction result. However, there are some methods that take into account more global information. For example, Node2Vec's [21] node embedding uses the node information as well as neighbourhood information when creating the topic embedding.

Not only in the context of ontology extension, the link prediction task is also applicable in the social network analysis [22], [23]. For instance, Murata and Moriyasu [24] used weighted proximity measures of the social networks alongside the social network's existing links' weights in Question-Answering Bulletin Boards to predict the evolution of social networks. Another application of the link prediction task in social media is shown in Papadimitriou *et al.* [25]. They proposed a faster and more accurate link prediction approach for friend recommendations. Fire *et al.* [26] proposed structural features that are simple and easy to compute for identifying missing links.

We argue that the extension of ontology works in a very similar fashion to the link prediction in the field of social network analysis. We consider ontology nodes as analogous to new information nodes in a social network. In social media, two users within the network may not be connected. However, by using link prediction methods, we can speculate if they will be connected in the future based on their similarities, for example, mutual friend(s), school, and location. This illustration can be applied to ontology as well—a link predictor can discover *hidden* links within an existing ontology, therefore, allowing an enrichment of the ontology with additional relationships. This is one of the aims of our study. We describe our approach of using the link prediction formulation for ontology extension in Section IV.

It has to be noted that our link prediction task is different from the typical text classification task. The text classification task normally uses a whole text document as input, while ontology terminologies contain only five-word terms at most.

G. WIKIPEDIA IN THE RESEARCH COMMUNITY

Wikipedia has been widely used in the research community due to its easily accessible data and having seemingly unlimited article collections. Different fields of research produced various methods of using Wikipedia in their respective studies. For example, Boyd [27] developed a grammatical error correction system for the German language using Wikipedia's revision history. An enriched Industry 4.0 dictionary[3] by Chiarello *et al.* [28] extracted important connections between technologies from Wikipedia. Their method also allowed for real-time updates to their dictionary.

The work of Kim *et al.* [29] is closely related to our study, where they constructed a technological ecology network using Wikipedia hyperlinks. They also applied link prediction methods for developing three predictive indicators of technological convergence. Still in the scope of Wikipedia for graph, Ni *et al.* [30] used Wikipedia alongside the Yahoo! knowledge graph to improve an entity recommendation task. On the other hand, Azad *et al.* [31] used Wikipedia and WordNet for Query Expansion.

In the context of this paper, we use Wikipedia to create a ground truth knowledge base (called the "Wikipedia Graph") for evaluating automatic ontology extension. This proposed evaluation procedure enables cheaper and faster assessment of automatic ontology extension approaches. Section VI-B5 explains this in more detail.

III. TARGET ONTOLOGY: THE COMPUTER SCIENCE ONTOLOGY

One of the contributions of this paper is an extended and enriched Computer Science Ontology (CSO) with up-to-date topics and relationships obtained from various classification techniques. The CSO itself is a topic ontology containing scientific topics of Computer Science publications. Salatino *et al.* [5] automatically generated the CSO by applying the Klink-2 algorithm [32] to the Rexplore dataset [33]. Despite being automatically generated, Salatino *et al.* [5] claimed that there are also manual revisions of the ontology's relationships which were performed by experts,[4] therefore, categorizing the CSO as a semiautomatically generated ontology.

A. STRUCTURE OF THE CSO

The CSO consists of several root topics including Computer Science, Linguistics, Geometry, and Semantics. Despite various root topics, the *main* root of their ontology is Computer Science. Salatino *et al.*'s [5] described the CSO as a large-scale *taxonomy* of research areas—this ontology's content is occasionally explained using taxonomy terms, e.g., children, parent, and descendants. Within the CSO, there are a total of 14,164 topics and 162,121 semantic relationships. Besides the "parent of" relationship, there are also other semantic relationships within the CSO.

[3]Industry 4.0 dictionary: www.industria40senzaslogan.it
[4]Please refer to http://cso.kmi.open.ac.uk/home

The CSO, like any other ontology, consists of triples: (1) source nodes (subjects) representing the parent topic, (2) predicates (pred.) representing the relationship between the source and target nodes, and (3) target nodes (objects) representing the direct child of the parent topic. In our study, we focus on the "*parent of*" relationship in the CSO. Table 1 illustrates the "parent of" relationship obtained from the CSO's website.[5] According to our own observation of the ontology, there are 32,043 links on the "parent of" relationship, and a total of 4,241 non-leaf topics that can act as potential parents for the link prediction task.

TABLE 1. **First five *"parent of"* predicates of the CSO's "computer science" topic.**

Subject	Pred.	Object
computer science	parent of	artificial intelligence
computer science	parent of	bioinformatics
computer science	parent of	computer aided design
computer science	parent of	computer hardware
computer science	parent of	computer imaging and vision
computer science	parent of	computer networks

For illustrative purpose of the actual format of the ontology, the CSO is shown in its website using a tree-structured visualization. However, in principle, its structure is a combination of a graph and a tree since some links do not follow the taxonomy constraint (refer to Figure 1). The concept of hierarchy exists, but a node can become a child of other nodes at deeper levels. This means, the *real* depth of a node is ambiguous. This situation often occurs at the lower (deeper) levels.

B. USING THE CSO AS DATASET

In this study, we mainly focus on the "parent of" relationship. The CSO's Computer Science root is used along with all of its direct children and each of their descendants. All topics in the CSO are used and processed in two different manners:

1) For the first variant, especially when using the Indirect approach, the ontology is automatically reformed in a way that is appropriate for the multilabel classification models by applying a **limitation rule**, which modifies the original format of the CSO into a more "flattened" version. Given a level limitation X, all descendants of a node A (A is at the level X) are all considered as its direct children, i.e., flattened as one level. While this procedure, unfortunately, removes some important details, it is necessary for machine learning purposes. The concept of the limitation rule is explained in more detail in Section IV-A1.
2) For the second variant, especially when using our proposed Direct approach, an **initial extension** of the CSO is given to the classification models. This is done by combining the CSO with a "Topic Graph". The steps taken to obtain the Topic Graph is explained in detail in Section V-B.

[5]http://cso.kmi.open.ac.uk/topics/computer_science

IV. BASELINE: INDIRECT APPROACH

The idea of the Indirect approach is to: (1) train a multilabel text classifier on data obtained purely from the seed CSO ontology and (2) select the best performing classifier to find the direct parent (supertopic) for a given new input topic, based on the characteristics of the seed ontology learned by the classifier.

Here, we adopt the classification framework proposed by Althubaiti *et al.* [19] for ontology extension. Previously, they showed the potential of a completely automatic ontology extension by applying a text classification method to extend a disease ontology. Figure 4 shows the "Flat Classification" Framework that has been adapted for our task. We first create topic vectors for terminologies in the ontology, and then feed these topic vectors to train machine learning classifiers. The following explains our implementation details.

A. SEED ONTOLOGY AND DATA PREPARATION

1) CLASSIFIER FRIENDLY ONTOLOGY FORMAT

During our previous experiment with the flat classification, method [20], the machine learning models performed poorly when using the original format of the CSO. We suspect there were too many potential parents (as many as 4,241 potential parents in the CSO). In addition, there are different numbers of nodes composing "subtrees", i.e., imbalanced data. For example, in several cases, a potential parent has hundreds of direct children, whereas other potential parents only have one child. This leads to an extremely skewed dataset.

To alleviate this problem, we propose to use a level limitation rule to increase the model performance, while conceding that this decision removes some detailed information. When we set the maximum level of potential parents to X, we only consider nodes from level 0 (root) to X as potential parents. Descendant nodes at level$\geq X+2$ are all collapsed as the direct children of the immediate topics at the level X; in other words, all nodes at level$\geq X + 2$ are moved to level $X + 1$. Figure 3 illustrates how the level limitation works when the limit is set to 2.

Each of the trees in Figure 3 is described as follows; First, Figure 3a illustrates the original format of the ontology. Here, the green nodes represent all potential target labels for the classifier. In the case of this study, nodes having at least one child are considered as parent nodes, i.e., classification labels. Second, in Figure 3b, the *level limitation rule* is applied. Here, the brown, orange, and lighter orange colors represent nodes at different levels, and the green color represents the nodes that are within the limitation rule, in this case limit $=$ 2. Third, Figure 3c shows the final form of the ontology after it is flattened based on the set *level limitation rule*. Here, the descendants of the node's second level nodes are all flattened and become their direct children.

A problem with using the CSO is extremely imbalanced data. For example, a parent node can have only one direct child, but a different parent node can have hundreds of direct children. By applying level limitation, we aim to decrease

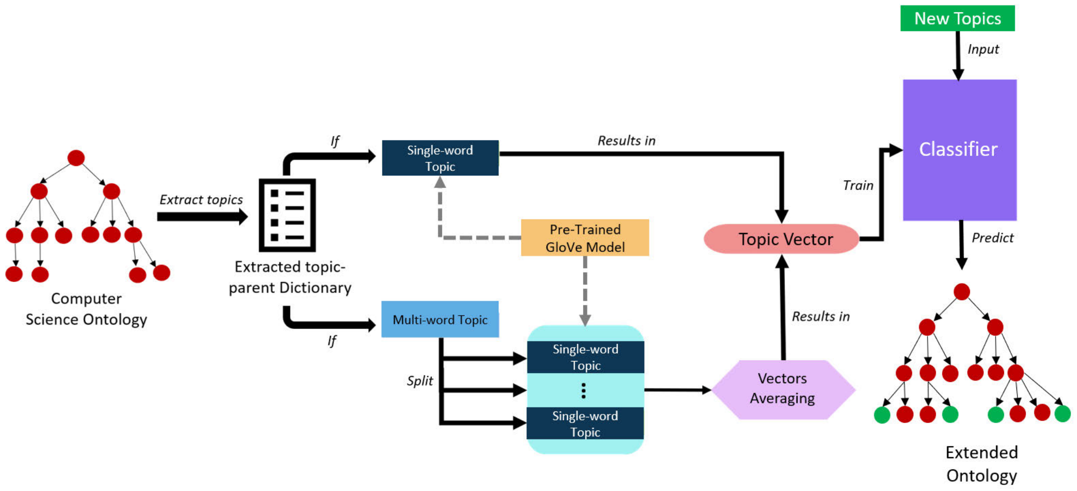

FIGURE 2. **Architecture of our flat classification framework.**

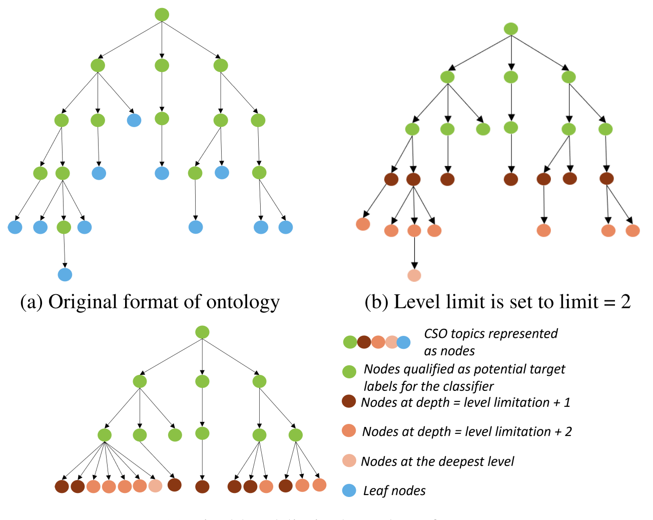
(a) Original format of ontology
(b) Level limit is set to limit = 2
CSO topics represented as nodes
Nodes qualified as potential target labels for the classifier
Nodes at depth = level limitation + 1
Nodes at depth = level limitation + 2
Nodes at the deepest level
Leaf nodes
(c) Final level-limited ontology format

FIGURE 3. **Level limitation rule process applied to the CSO with limit = 2.**

the imbalance, and thus improve the performance of the classifier.

Next, we convert the CSO as a ‘‘topic dictionary’’, consisting of a list of links between object nodes (children) and their corresponding subjects (parents). Before applying the level limitation rule to the CSO, the topic dictionary consists of 32,043 links with 4,241 potential target parents. After applying the level limitation rule, the links in the topic dictionary decreased to 18,497 links with 31 potential target parents and 15,748 links with 20 potential target parents. Instances (object–subject pairs) in this topic dictionary are then used to train and test machine learning models. We represent the nodes as topic embeddings. In this paper, we report our expeirment with the three best performing algorithms: *Gaussian Naive Bayes*, *Support Vector Machine* and *Multilayer Perceptron* algorithms, even though we experimented with more algorithms.

2) TOPIC EMBEDDINGS

A terminology (topic) may contain one or more words. A one-word topic is represented as a single vector embedding. On the other hand, a multi-word topic is first split into its individual words, and represented as the average of the individual word embeddings (cf. Figure 2). We use a pretrained GloVe [34] model[6] to obtain the topic representations.

V. PROPOSED METHOD: DIRECT APPROACH

In this paper, we propose a Direct approach that is the opposite of the Indirect approach. Figures 4 illustrates the first part of our Direct approach. Here, hundreds of new Computer Science topics are first obtained from paper abstracts (‘‘Topic Graph’’ in the Figure). We then directly connect them to the seed CSO ontology through intersecting topics, resulting in a combined CSO-Topic graph. We consider this an *initial extension* of the ontology. This undoubtedly results in numerous noisy links. To this end, we use a classifier to *filter* out noisy links and predict hidden links—this differentiates the role of the classifier from the previous Indirect approach. We obtain the topic vectors for training and testing examples via Node2Vec embeddings trained on the initially extended ontology. In this way, we *directly* involve the new topics in the node vector construction process.

We experiment with several classification models, which are categorized as the flat classification techniques in the Indirect approach since node-level information is not considered. The reason is that by initially extending the CSO Graph, the level information has been implicitly taken into consideration. Section V-D explains this in more detail.

[6]840B pre-trained GloVe vectors on Common Crawl. https://stanford.io/3cAbqY5

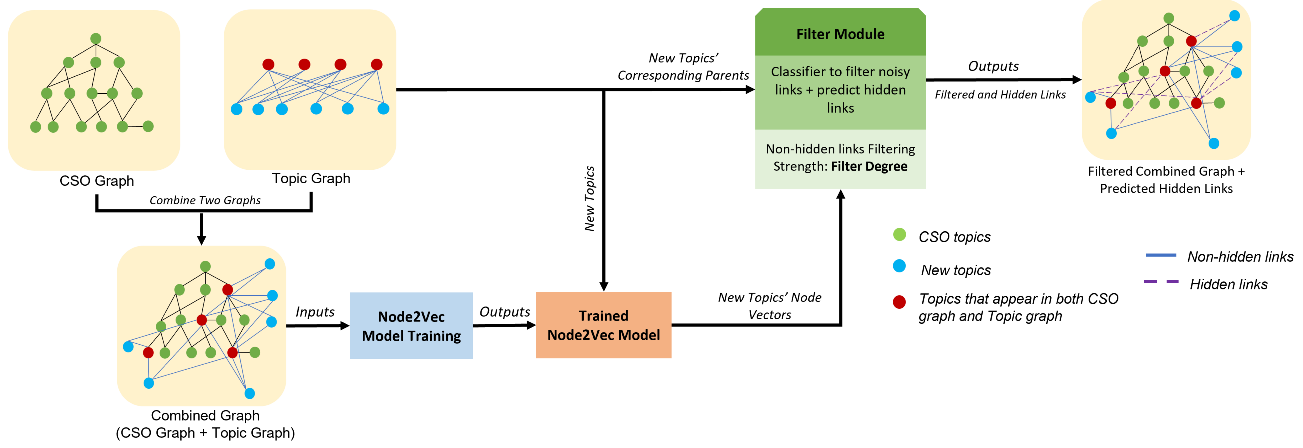
CSO Graph
Topic Graph
Combine Two Graphs
Combined Graph
(CSO Graph + Topic Graph)
Inputs
Node2Vec Model Training
Outputs
Trained Node2Vec Model
New Topics
New Topics' Corresponding Parents
New Topics' Node Vectors
Filter Module
Classifier to filter noisy links + predict hidden links
Non-hidden links Filtering Strength: Filter Degree
Outputs
Filtered and Hidden Links
Filtered Combined Graph + Predicted Hidden Links
CSO topics
New topics
Topics that appear in both CSO graph and Topic graph
Non-hidden links
Hidden links

FIGURE 4. Process of the direct approach's main module, the filter module.

The initial performance of the *filtering* step is measured by a *filter degree*, followed by further evaluation using the Wikipedia-graph based precision, recall, and F1 metrics (more on our evaluation procedure in Section VI-B3). The following subsections explain our data preparation methods and the details of each step in our approach.

A. EXTRACTING NEW TOPICS FROM PAPER ABSTRACTS

To obtain as many new Computer Science topics as possible, we scrape a total of 5,798 paper abstracts from multiple proceedings in the ACM digital library.[7] We use recent papers that were published from 2018 to 2020 (we also occasionally take papers published in 2021 if available). From each of the scraped paper abstracts, we extract its *title*, *abstract*, and *tags* information. We call these types of information as the *sources* of either new topics or CSO topics. Out of these sources, the tags information is the main source used to obtain the actual new topics for extending the CSO, while the other sources are used to obtain topics that already exist in the CSO. This action helps us establish as many *initial* connections between the new topics and the CSO. Section V-B explains how we extract the new topics.

B. TOPIC GRAPH

For simplicity, we refer to the topics that have already existed in the original CSO as *CSO topics*, while the term *new topics* is used to refer to the topics that do not originally exist in the CSO. As has been previously explained, the new topics are obtained from recently published papers' tags information. In the ACM Digital Library, each paper is accompanied by a webpage. Here, the authors provided manually inputted tag information on the keywords of their studies. Therefore, we can ensure the correctness of the tag information, as opposed to using a term extractor which may generate unnatural outputs (cf. Section II-C).

[7] ACM digital library, https://dl.acm.org/

We then use abstracts and titles to find the existing CSO topics that may appear in the paper together with the new topics. Once we find the co-occurrence of existing CSO topics and new topics in the same paper, we connect them to build a sub-topic graph (a bipartite graph, one sub-topic graph for one paper). Figure 5 shows an example of a sub-topic graph. We then combine sub-topic graphs from multiple papers as a single, complete, Topic Graph.

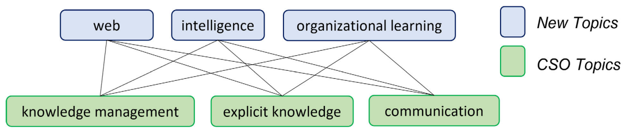

FIGURE 5. Format of the sub-topic graph.

Figure 6 illustrates the next step to initially extend the Computer Science Ontology using the Topic Graph. The idea is to simply combine the Topic Graph and the CSO Graph via mutual topics. This is easy to do since the Topic Graph already contains connections between new topics and existing CSO topics. The combined graph is then used to train topic vectors via Node2Vec.

C. TOPIC GRAPH's TOPIC DICTIONARY

Similar to the topic dictionary used in the Indirect approach, we also build a topic dictionary to connect (link) new topics and their corresponding parents in the CSO in the Direct approach. The topic dictionary is built based on the Topic Graph. Therefore, instead of using all connections within the original CSO, only the CSO topics that are confirmed to be related to the new topics are used as the classification labels. This strategy significantly decreases the number of potential labels compared to the Indirect approach, which uses all CSO topics as potential labels. In the Indirect approach, there are 4,241 potential labels, while only 1,991 potential labels exist for the Direct approach, based on the original version of the CSO without level limitation.

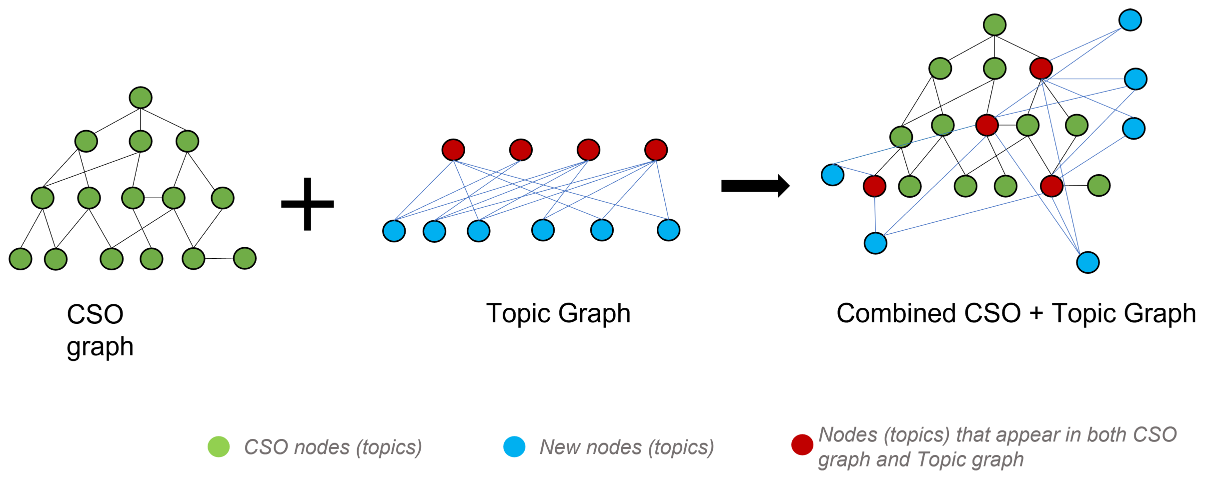
CSO graph
Topic Graph
Combined CSO + Topic Graph
CSO nodes (topics)
New nodes (topics)
Nodes (topics) that appear in both CSO graph and Topic graph

FIGURE 6. Illustration of initial extension of the computer science ontology with new topics obtained from newly published papers.

D. IMPLICIT LEVEL INFORMATION

As shown in Figure 6, combining the Topic Graph and the CSO Graph involves placing the new topics within the appropriate levels under their corresponding CSO supertopics. Thus, we implicitly consider the node-level information in the Direct approach even when we only implement flat classification models in the subsequent filtering step.

E. FILTERING STEP

The filtering step is the most important step in our Direct approach. Given a new topic, a multilabel classifier decides its supposed CSO parent(s). The filtering step has two roles: *to filter out noisy links* and *predict hidden links*. Figure 4 illustrates the action of the filter module. The input to the filter module is a set of node vectors obtained from the combined CSO-Topic Graph alongside each of the node's supposed parent(s). The output of this module is the combined CSO-Topic Graph that has been filtered, consisting of the remaining valid links and predicted hidden links.

1) FILTERING NOISY LINKS

Given a link between the new topic and existing CSO topic, we use the classification output to judge whether the link as noise or not. If the classifier does not recognize the presence of a link between a new topic and a parent CSO topic, then we assume that the corresponding link as noise. Subsequently, we remove it from the combined CSO-Topic Graph. We experiment with several classification algorithms and select the best one based on how well it can filter out the noisy links. An ideal model should not filter out too many or too few links. This can be evaluated using the *filter degree* measure (cf. Figure 4), which indicates the percentage of links filtered out by a chosen classifier. An ideal model should also have the ability to recognize semantically valid links. We evaluate this aspect based on the Wikipedia-based F1-score explained in Section VI.

2) PREDICTING HIDDEN LINKS

Aside from filtering out noisy links from the combined CSO-Topic graph, the filter module can also predict hidden links between topics in the combined CSO-Topic Graph. This is to take into account potential connections between one topic to another topic that appear in entirely different source papers. Figure 7 illustrates this. We then determine the validity of the predicted hidden links by comparing them with the links in the Wikipedia graph (cf. Section VI-B4). Section VI-B3 describes how we implement this. Figure 8 shows the relationships among various types of links that appear in our study.

VI. EXPERIMENTS AND RESULTS

In this study, we use the latest version (at the time of this study) of the CSO (v.3.2). The CSO is used as a seed ontology for both the Indirect and Direct approaches. Slight modifications are done to the original CSO format, specifically for the Indirect approach where the level limitation rule is applied (we set level limitation = 1).

A. EXPERIMENTAL SETTING: INDIRECT APPROACH

1) DATA PREPARATION

Inspired by the work of Athubaiti *et al.* [19], we adapt and implement a multilabel topic classifier trained on the CSO. The topic vectors are used as training and test instances. Given a topic, we feed the classifier with its direct parent(s) as labels. Table 2 shows an example.

TABLE 2. A sample of the data format. "Vector_rep" is the 300 dimension vector representation of the "Topic."

Topic	Vector_rep	Supertopics
artificial intelligence	-0.744,..., 0.169	[computer science]
service robots	-0.394,...,0.306	[intelligent robots, robots]

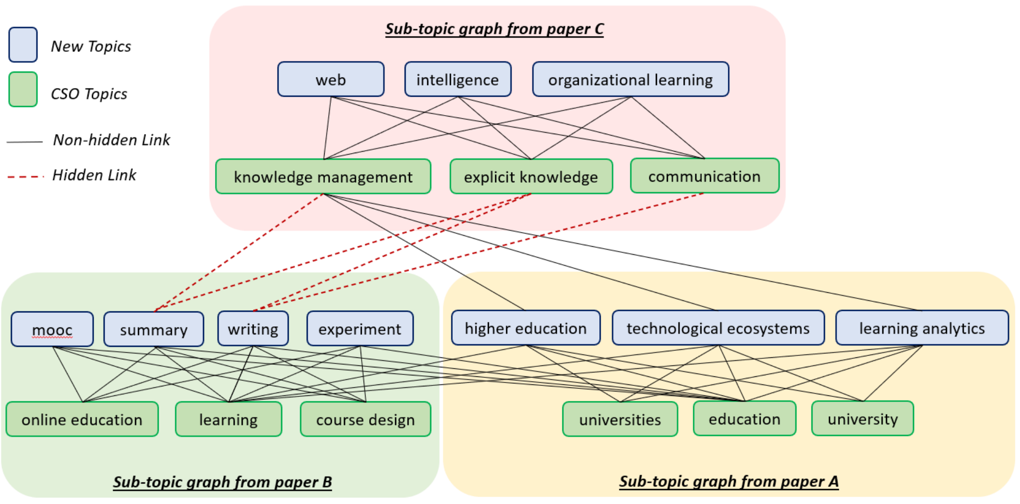

FIGURE 7. Illustration of hidden links in the topic graph.

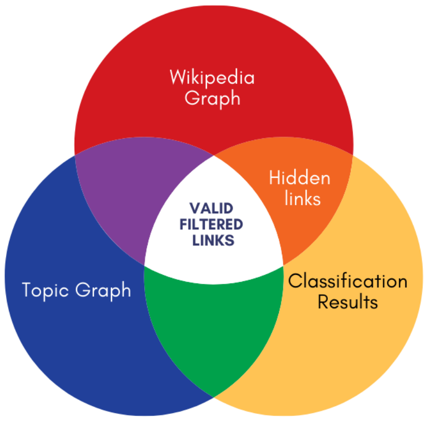

FIGURE 8. Relationship between the links in the Topic Graph, Wikipedia graph, and classifier's predicted links.

We consider three types of data treatment in this study: one using the original structure of the CSO and applying two level-limitation rules (level limitation=1 with and without synonyms) to the CSO. There are 4,241 potential parents (classes) when using the original structure of the CSO. However, we decreased the number of potential parents to only 31 and 20 when applying the level-limitation rule, with and without using synonyms respectively.[8] Another reason for applying the level-limitation rule is to compare our work with the work in [19] and make a similar number of parents to the approach in [19] since Althubaiti *et al.* [19] uses a maximum of only 17 potential parents.

[8]In the CSO, terminologies sometimes have synonyms, such as the plural version of a topic. These synonyms may have different direct children. When not using synonyms, we collapse their direct children.

2) EVALUATION METRICS

Most classification studies evaluated their models' performance using standard *precision*, *recall*, and *F1 score* metrics. However, this is not applicable in our case for two reasons. First, the Topic Graph in the Direct approach contains noisy links. Therefore, it is necessary to first determine the validity of the links. Second, the multilabel classification task has more than one gold label. Exact matching to the set of gold labels is difficult due to the number of classes. In this study, *we evaluate the multilabel classification performance as if it was a binary classification task by using Wikipedia-based evaluation metrics*. This is analogous to evaluating partial multilabel matches.

To have a fair comparison to the Direct approach, a specific evaluation procedure is also devised for the Indirect approach. The Indirect approach's classification models classify a set of entirely new topics obtained from the Topic Graph, and the resulting link predictions are then validated using the Wikipedia-based performance metrics. Section VI-B5 explains these metrics in detail.

3) CHOICE OF CLASSIFICATION ALGORITHMS

For the Indirect approach, we initially implement several machine learning models to "replicate" the results of Althubaiti *et al.* We used *Logistic Regression* (LR), *Gaussian Naive Bayes* (GNB), *K Nearest Neighbor* (KNN), *Decision Tree* (DT), *Random Forest* (RF), *Support Vector Machine* (SVM), and several versions of *Multilayer Perceptron* (MLP). Table 3 shows the experimental result when the models were evaluated using standard *precision*, *recall* and *F1 macro* scores. However, the models in our Direct approach cannot be evaluated using the standard measures due to the reasons explained in Section VI-A2. Therefore, we only report Table 3 strictly for general interests.

TABLE 3. Performance of flat classification models in the Indirect approach with data types. "Limit1_19" indicates the models' performance when applying the level limitation rule =1 without synonyms to the dataset. "Limit1_30" shows the performance when applying the same level limitation rule with synonyms. "No_limit" shows the performance when the original CSO data does not undergo any changes. Best performance is written in boldface.

Model	CSO Versions								
	Limit1_19			Limit1_30			No_limit		
	precision	*recall*	*F1-macro*	*precision*	*recall*	*F1-macro*	*precision*	*recall*	*F1-macro*
LR	**0.451**	0.194	0.264	**0.338**	0.126	0.179	0.010	0.005	0.006
KNN	0.427	0.220	**0.279**	0.300	0.141	0.187	0.007	0.003	0.004
DT	0.147	0.112	0.123	0.099	0.068	0.079	0.007	0.005	0.005
RF	0.124	0.048	0.066	0.073	0.024	0.035	0.000	0.000	0.000
GNB	0.171	**0.650**	0.242	0.125	**0.648**	0.189	**0.022**	**0.163**	**0.032**
SVM	0.446	0.209	0.272	0.324	0.119	0.167	0.018	0.014	0.014
MLP_10	0.280	0.241	0.254	0.215	0.172	**0.189**	0.002	0.001	0.001
MLP_50	0.315	0.238	0.261	0.206	0.165	0.180	0.000	0.000	0.000
MLP_100	0.284	0.236	0.254	0.212	0.164	0.180	0.000	0.000	0.000
MLP_200	0.295	0.238	0.260	0.217	0.165	0.183	0.000	0.000	0.000

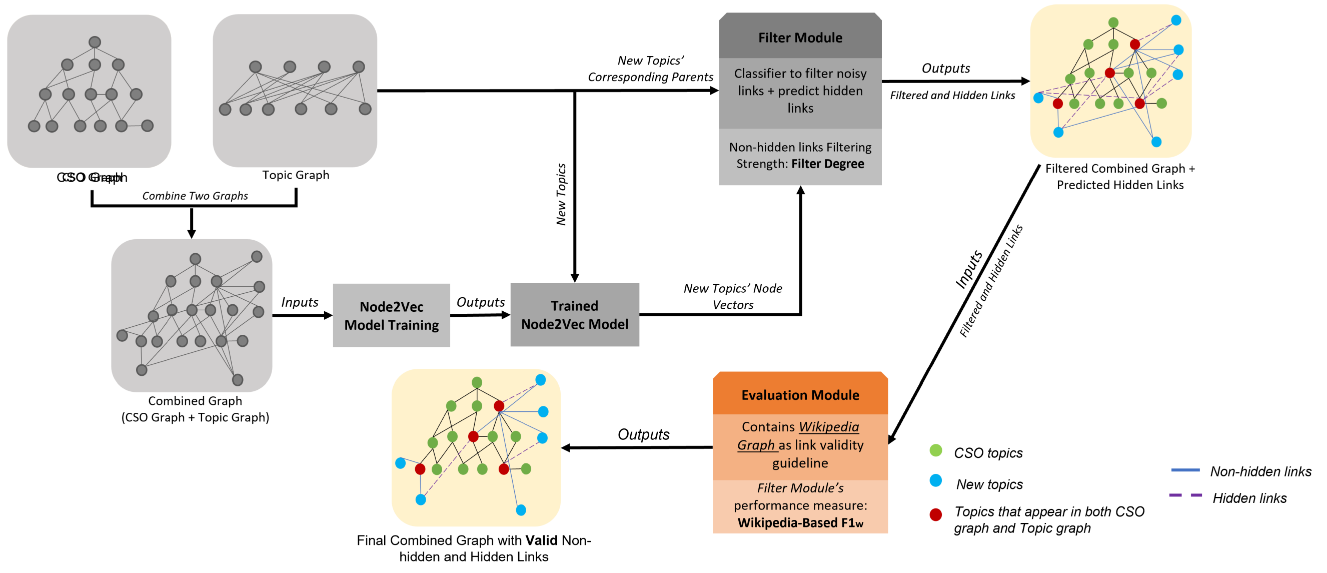
CSO Graph
Topic Graph
Combine Two Graphs
Combined Graph
(CSO Graph + Topic Graph)
Inputs
Node2Vec Model Training
Outputs
Trained Node2Vec Model
New Topics
New Topics' Corresponding Parents
New Topics' Node Vectors
Filter Module
Classifier to filter noisy links + predict hidden links
Non-hidden links Filtering Strength: Filter Degree
Outputs
Filtered and Hidden Links
Filtered Combined Graph + Predicted Hidden Links
Inputs
Filtered and Hidden Links
Evaluation Module
Contains Wikipedia Graph as link validity guideline
Filter Module's performance measure: Wikipedia-Based F1w
Outputs
Final Combined Graph with Valid Non-hidden and Hidden Links
CSO topics
New topics
Topics that appear in both CSO graph and Topic graph
Non-hidden links
Hidden links

FIGURE 9. Process of the direct approach's evaluation module.

In the following, we only take the best performing models and discuss the performance of GNB, SVM, MLP (using 10 (MLP_10), 50 (MLP_50), 100 (MLP_100), and 200 (MLP_200) neurons) under both Indirect and Direct approaches. On top of that, we use our proposed Wikipedia-based evaluation metrics for a fair comparison.

B. EXPERIMENTAL SETTING: DIRECT APPROACH

1) TRAINING AND TESTING DATA PREPARATION

To obtain the topic node representation, a Node2Vec model is trained on the combined CSO-Topic Graph, and later used to extract the vector representation of the new topics' position within its neighborhood and its neighborhood information in the graph. Given a new topic vector, a link prediction model predicts the potential CSO topics that should be set as its parents (i.e., classification labels).

2) FILTER MODULE EVALUATION

Similar to the Indirect approach, we implement the filter module using GNB, SVM and four MLP variants. We evaluate the performance of our filter module using an ad-hoc metric called the *filter degree*. It denotes the percentage of links that are removed from the Topic Graph. A moderate score is better for this metric. However, extreme numbers, such as 100% or 0%, are undesirable since the links are all filtered out or remain. This means that the filter module does not serve its purpose. Equations 1 and 2 show the formulas to calculate the *filter degree*. "Noisy link true positives" indicate the noisy links predicted by classifier, while "test data" indicates the links that exist in the test data portion of our Topic Graph.

$$\text{filter degree} = 1 - \frac{|\text{noisy link true positives}|}{\text{number of test data}} \tag{1}$$

$$\text{noisy link true positives} = \text{predicted links} \cap \text{test data} \tag{2}$$

3) WIKIPEDIA-BASED EVALUATION PROCEDURE

In the Direct approach, we also evaluate the quality of the final output graph resulting from the filter module by comparing it to the links in the Wikipedia graph. Figure 9 illustrates this. Here, there are two types of links that are considered as *valid links*. First, valid *non-hidden* links which are the remaining links in the filtered graph that exist within both the Topic Graph and Wikipedia graph. These valid *non-hidden* links are referred in Figure 8 as "valid filtered links." Second, the valid *hidden* links—the remaining links in the filtered graph that do not exist within the Topic Graph but exist within the Wikipedia graph. Our Wikipedia-based evaluation procedure is built only on the performance of valid filtered links.

4) WIKIPEDIA GRAPH

The Wikipedia Graph is the ground truth used in the last step of the Direct approach. We list unique new topics and CSO topics in the Topic Graph (cf. Section V-B), and use them as search queries on the Wikipedia API. The Wikipedia API then returns a list of titles of related articles from the given search query. We apply the following two rules before proceeding to the next step:

1) If there is *no exact match* within the returned list of Wikipedia article titles (search results) to the search query, we extract the complete Wikipedia articles from all of the search results.
2) If an *exact match exists* within the returned list of search results, we only extract one complete Wikipedia article that has the same title as the search query.

After obtaining all of the needed Wikipedia Articles, important topic terms from each of these articles are extracted using the YAKE keyword extraction tool [11]–[13]. All of the obtained keywords are then linked to the search query as ground truth.

5) WIKIPEDIA-BASED EVALUATION METRIC

Along with using the *filter degree*, we also check the validity of the links in the filtered graph based on their presence in the Wikipedia Graph. We propose ad-hoc evaluation metrics called the Wikipedia-based $precision_w$ (Equation 4), $recall_w$ (Equation 5), and $F1_w$ (Equation 6). However, before proceeding with their formulas, we first explain *valid filtered links* as follows.

The performance of the filter module is based on how many *valid filtered links* are produced by the multilabel classifier. To obtain the *valid filtered links* (Equation 3), the *noisy link true positives* obtained from Equation 2 should be intersected again with the set of Wikipedia graph links (WG links).

$$\text{valid filtered links} = \text{noisy link true positives} \cap \text{WG links} \quad (3)$$

In the case of $precision_w$ (Equation 4), the *valid filtered links* are divided by the number of links within the set of *predicted links* obtained from a classifier model in the filter module. For calculating $recall_w$ (Equation 5), the number of *valid filtered links* is divided by the number of valid links within the test data—the intersection of the test data with WG links.

$$precision_w = \frac{|\text{valid filtered links}|}{|\text{predicted links}|} \quad (4)$$

$$recall_w = \frac{|\text{valid filtered links}|}{|\text{test data} \cap \text{WG links}|} \quad (5)$$

$$F1_w = 2 \times \frac{precision_w \times recall_w}{precision_w + recall_w} \quad (6)$$

The final performance score of a classifier is then decided by the $F1_w$ (Equation 6) score. However, aside from the links limited only to the Topic Graph, *valid remainder prediction results* are also taken into consideration. *Valid remainder links* are contained in a set of links predicted by the classifier that *appear in the Wikipedia Graph but do not appear in the Topic Graph*. In other words, the *valid remainder links* are the same as the *hidden links* explained in Section V-E2. Refer to Figure 7 for an illustration.

C. EMPIRICAL RESULTS

In this section, we compare the performance of the Direct and Indirect approaches based on their filtering ability, Wikipedia-based $F1_w$ score, and the number of new links and topics predicted. For a fair comparison, the classification models in both approaches are trained using the same complete data obtained from the Direct approach's Topic Graph. We perform the experiment using a 5-fold cross validation method, with 60% of the data set used as training data and the remaining 40% used as testing data, and report the average performance. We use three types of treatment to the CSO data: CSO with level limit = 1 without including synonyms ("limit1_19"), CSO with level limit = 1 with synonyms "limit1_30", and CSO without level limit (original format of CSO, "no_limit").

We compare two types of classifiers: (1) classifiers trained as *topic classifiers* from the Indirect approach and (2) classifiers trained as *link predictors* from the Direct approach. In practice, we use ten classification models. However, we only show the six best performing models, i.e., GNB, SVM, and four versions of MLP for evaluation and performance comparisons.

1) FILTER DEGREE

Here, we consider the role of classifiers as filters for noisy links (as explained in Section V-E1). The filter degree represents the ability of a classifier to keep the valuable links between two topics and remove noisy links. Table 4 shows the filter degree score of each model.

Table 4 shows that the classifier models are better at picking good links and removing noisy links when trained under our Direct approach. Classifiers in the Indirect approach filtered out more than 98% of the links within the Topic Graph. This is not desirable as it is essential to retain correct links in the Topic Graph and only selectively remove the noisy ones.

TABLE 4. Filter Degree score comparison between models of the two approaches: Indirect and direct.

Model	limit1_19		limit1_30		no_limit	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	0.995	0.246	0.994	0.246	0.988	0.245
SVM	0.999	0.165	0.999	0.164	0.999	0.178
MLP_10	0.999	0.190	0.999	0.190	1.000	0.200
MLP_50	0.999	0.227	0.999	0.227	1.000	0.238
MLP_100	0.999	0.237	0.999	0.237	1.000	0.247
MLP_200	0.999	0.243	0.999	0.242	1.000	0.254

TABLE 5. The $F1_w$ score for models of the two approaches: Indirect and Direct. Best scores are shown in boldface.

Model	limit1_19		limit1_30		no_limit	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	**0.032**	0.274	**0.034**	0.273	**0.077**	0.273
SVM	0.004	0.267	0.003	0.267	0.002	0.267
MLP_10	0.004	0.271	0.004	0.271	0.001	0.270
MLP_50	0.004	0.275	0.002	0.276	0.001	0.275
MLP_100	0.005	**0.278**	0.004	0.278	0.001	**0.277**
MLP_200	0.006	**0.278**	0.002	**0.279**	0.000	**0.277**

2) WIKIPEDIA-BASED PERFORMANCE SCORES

Table 5 shows the performance based on the Wikipedia $F1_w$ scores, where the classifier models under the Direct approach performed better than the models in the Indirect approach by a large margin. For the Indirect approach, the GNB model achieves the best $F1_w$ scores on all data treatment variations. The GNB algorithm also performs well in the Direct approach, showing close-to-the-highest scores consistently. Tables 6 and 7 show the $precision_w$ and $recall_w$ scores, respectively. In Table 6, the Indirect approach generally attains a greater precision compared to the Direct approach. However, its $recall_w$ scores shown in Table 7 are very low. In real-world applications, a higher recall score is also important to produce more valid links for the final extension of the ontology.

TABLE 6. The $precision_w$ score for models of the two approaches: Indirect and direct.

Model	limit1_19		limit1_30		no_limit	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	0.509	0.164	0.474	0.164	0.527	0.164
SVM	0.272	0.157	0.358	0.157	0.202	0.157
MLP_10	0.266	0.160	0.264	0.160	0.001	0.160
MLP_50	0.266	0.164	0.138	0.165	0.200	0.164
MLP_100	0.276	0.166	0.286	0.166	0.200	0.165
MLP_200	0.310	0.166	0.196	0.167	0.133	0.166

TABLE 7. The $recall_w$ score for models of the two approaches: Indirect and direct.

Model	limit1_19		limit1_30		no_limit	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	0.017	0.837	0.017	0.837	0.042	0.836
SVM	0.002	0.890	0.890	0.001	0.001	0.877
MLP_10	0.002	0.882	0.002	0.880	0.001	0.868
MLP_50	0.002	0.860	0.001	0.863	0.000	0.849
MLP_100	0.003	0.859	0.002	0.858	0.000	0.845
MLP_200	0.003	0.855	0.001	0.855	0.000	0.839

We also selected the best performing approach and models based on the number of actual valid links and topics produced by the model. The following section (Section VI-C3) provides more information on this issue.

3) POTENTIAL NEW LINKS AND NEW TOPICS

This section presents the evaluation based on the number of potential new links and topics resulting from classifier models (i.e., the filter module). We consider two types of predicted links: (1) *non-hidden links*—the predicted links that also appear in the Topic Graph, and (2) *hidden links*—the predicted links that do not appear in the Topic Graph. Furthermore, we also include the number of total links produced by each of the classifiers. *Total links* is the sum of non-hidden and hidden links. We consider an ideal model as a model that is able to produce a high amount of total links. The number of total links is essential in this study because aside from the Wikipedia-based $F1_w$ score, we are using it to support our conclusion regarding the best performing model. In the end, we expect the best performing model to have the most potential for production due to its ability of producing numerous valid links.

TABLE 8. Approach comparison: Average valid non-hidden links. Best scores are shown in boldface.

Model	limit1_19		limit1_30		no_limit	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	**23.2**	1179.2	**24.8**	1178.6	**58.8**	1178.0
SVM	2.2	**1254.2**	1.8	**1254.2**	1.2	**1234.6**
MLP_10	3.2	1241.6	2.8	1239.2	1.0	1223.0
MLP_50	3.2	1121.0	2.0	1215.0	0.4	1195.0
MLP_100	3.6	1208.8	0.4	1190.2	4.2	1203.8
MLP_200	4.2	1203.8	1.8	1205.0	0.4	1181.8

Table 8 presents the performance based on valid non-hidden links for each classification model, while Table 9 presents the performance based on valid hidden links. GNB performs the best for the Indirect approach on both aspects. We can see that the highest number of both non-hidden and hidden links is achieved when the level limitation rule is not applied. This is good news since we lose fine-grained information when applying the level limitation rule. However, no matter how good the performance of the GNB algorithm under the Indirect approach is, its performance is still notably worse when compared to the hidden links resulted by the same GNB model under the Direct approach.

The Indirect approach results in near-zero number of links in average for the valid non-hidden links (except for the GNB model), and a completely zero number of valid hidden links. We suspect that this is caused by the coarse-grained results given by the topic classifiers.

The topic classifiers are trained only on CSO data (without new topics); thus, the classifiers are limited to only 20 (for limit1_19) and 31 (for limit1_30) target classes which are located at the first and root level of the CSO. These

TABLE 9. Approach comparison: Average valid hidden links. Best scores are shown in boldface.

Model	CSO Versions					
	limit1_19		**limit1_30**		**no_limit**	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	0.0	**416.8**	0.0	**420.4**	0.0	**538.0**
SVM	0.0	23.2	0.0	23.8	0.0	32.4
MLP_10	0.0	25.8	0.0	29.0	0.0	36.4
MLP_50	0.0	29.2	0.0	27.2	0.0	36.6
MLP_100	0.0	28.0	0.0	27.4	0.0	33.0
MLP_200	0.0	30.4	0.0	29.2	0.0	35.4

TABLE 10. Approach comparison: Average totals of valid links. Best scores are shown in boldface.

Model	CSO Versions					
	limit1_19		**limit1_30**		**no_limit**	
	Indirect	*Direct*	*Indirect*	*Direct*	*Indirect*	*Direct*
GNB	**23.2**	**1596.0**	**24.8**	**1599.0**	**58.8**	**1716.0**
SVM	2.2	1277.4	1.8	1278.0	1.2	1267.0
MLP_10	3.2	1267.4	2.8	1268.2	1.0	1259.4
MLP_50	3.2	1240.2	2.0	1242.2	0.4	1232.4
MLP_100	3.6	1238.2	2.6	1236.2	0.4	1223.2
MLP_200	4.2	1234.2	1.8	1234.2	0.4	1217.2

too-general links resulted from the classifiers may not exist in the Wikipedia Graph, where connections between topics are more specific (fine-grained). Since we decide the validity of a link through its existence within the Wikipedia Graph, the majority (if not all of the links) are automatically considered as not valid. Despite the majority of the resulted links being filtered out, we did not find this problematic due to the fact applying level limitation rule is undesirable in real-world applications. Additionally, text classification models typically do not perform well when they are trained on data with thousands of target classes (in the case of "no_limit"). This common limitation of machine learning approach also possibly causes the low performance scores of the Indirect approach.

The limitations of the Indirect approach, however, do not apply to the Direct approach. This is because we train the classification models using the CSO data that has been initially extended with the Topic Graph. This procedure allows the consistency of initial connections between new topics and their correct CSO topics, no matter how the dataset is treated. As the result, the final resulting links given by the Direct approach are similar, despite differences in treatment to the CSO.

We next take a look at the model performance based on the number of total valid links produced. Table 10 shows the evaluation result. Here, the GNB model produces the highest number of total links under both the Indirect and Direct approaches. This table suggests that GNB is the most reliable model, and that the Direct approach is the most reliable approach for real-life deployment.

We can obtain two types of new topics from the predicted links: (1) new topics obtained from the valid non-hidden links, and (2) new topics obtained from the valid hidden links. Table 11 shows the number of new topics obtained from the resulting non-hidden links. Consistent with the previous results, the GNB attains the best performance under the Indirect approach. While the best performing model for the Direct approach is the SVM.

A similar situation also applies to the number of topics obtained from the resulting hidden links. Table 11 shows the number of new topics resulted from the predicted hidden links. Here, GNB shows the best performance under both the Indirect and Direct approaches. Even though the GNB results in fewer non-hidden topics than the SVM, GNB is generally a more preferable choice in the Direct approach since it produces a notably higher number of hidden topics than the SVM model. In turn, this drastically increases the number of final new topics that would be put into the final extended ontology.

To appreciate the quality of our models, Table 12 shows eight randomly sampled new topics from each model of the Direct approach (from the 5th fold of the 5-fold-cross-validation with the no_limit CSO version) and their respective predicted parent topics in the CSO. In the real-world application, these new topics will be placed directly under their respective CSO supertopics for the CSO extension.

4) SIGNIFICANCE TESTING USING PAIRED t-TEST

A paired t-test is usually used when there is an interest in the difference between two variables under the same condition. Dietterich [35] proposed a version of the paired t-test called 5×2cv t-test for observing two machine learning classification models which seemingly have little difference in their performance scores, but may produce highly different results in reality. The 5×2cv paired t-test splits the dataset into two equal parts (50% and 50%) for training and testing each, and this action is repeated five times in total. In other words, "5×2cv paired" means a 2-fold cross-validation done five times.

Here, we perform the aforementioned statistical test on the top three performing models in the Direct approach, which are selected based on the number of total resulting links (as shown in Table 10). According to the total links, GNB is the best performing model, followed by SVM, with MLP_10 at last. Table 13 shows the resulting p-values of every model pair. Here, the null hypothesis is that the difference in performance between two models is not significant. According to the p-values shown in Table 13, we can reject the null hypothesis. This means that even though the classification models have little difference in performance score, their behavior is different. On the other hand, the statistical test is unnecessary for the Indirect approach since the best-performing GNB outperformed all other models by a large margin.

D. MANUAL EVALUATION

On top of the empirical evaluation, we additionally employ four human judges (four professors in Computer Science) to manually assess the quality of classification results. Recall that the classifiers under our Direct approach produced

TABLE 11. Approach comparison: average number of topics from valid non-hidden links and valid hidden links. Best scores are shown in **boldface**.

Model	CSO Versions											
	Limit1_19				*Limit1_30*				*No_limit*			
	Indirect		*Direct*		*Indirect*		*Direct*		*Indirect*		*Direct*	
	Non-hidden	Hidden	Non-hidden	Hidden	Non-hidden	Hidden	Non-hidden	Hidden	Non-hidden	Hidden	Non-hidden	Hidden
GNB	**23.2**	0.0	766.2	**278.4**	**24.8**	0.0	767.0	**284.4**	**52.0**	0.0	765.6	**338.0**
SVM	3.2	0.0	**805.8**	22.6	1.8	0.0	**804.8**	59.0	1.2	0.0	**800.6**	31.4
MLP_10	3.0	0.0	799.6	24.6	2.8	0.0	800.2	28.2	1.0	0.0	796.6	34.2
MLP_50	3.4	0.0	790.2	28.0	2.0	0.0	792.8	25.6	0.4	0.0	787.2	35.0
MLP_100	3.4	0.0	789.6	26.8	2.6	0.0	789.2	25.6	0.4	0.0	786.8	33.4
MLP_200	4.2	0.0	785.0	28.8	1.8	0.0	787.4	27.4	0.2	0.0	780.4	34.2

a promising number of average new links and topics in Section VI-C3 for ontology extension. However, the Topic Graph and Wikipedia Graph were automatically generated. Therefore, it is natural to doubt the *actual* quality of our systems. Here, the manual evaluation acts as an additional quality assurance. On top of that, it also serves as a meta-evaluation of the reliability of our Wikipedia-based evaluation procedure—whether the rank of top performing systems (GNB, SVM, MLP_10) selected by our automatic evaluation procedure are aligned with the rank formed from the opinions of human judges.[9]

In the manual evaluation, we ask judges for their opinions regarding the connections between two topics. Given several sample output links from each of the selected classification models, evaluators judge whether each link is "valid" (true) or "not valid" (false). Two types of links are used here: 30 randomly selected *hidden links* and 60 randomly selected *non-hidden links*. The difference in the number of links (30 hidden links and 60 non-hidden links) is due to the number of links produced by the classifiers. In the case of hidden links, the classifiers, aside from GNB, could only produce an average of 30 hidden links.

We describe the flow of our manual evaluation as follows. First, we check whether the output links (deemed as *valid* (true) links by classification models) are also deemed as valid by expert judges. If judges highly agree with automatic outputs, it means that we can believe the capability of the Wikipedia-based metrics as an evaluation metric. We show such statistics using an ad-hoc metric called "*evaluator-classifier ratio*" (ECR) between evaluator-classifier pairs. Second, we consider the potential issue of human inconsistency. Different experts might have different opinions even on the same subject matter. To this end, we calculate the inter-annotator agreement between evaluators (judges) to show the reliability of a model. Here, we assume that the outputs from a "good" model would be agreed by many human annotators with a high degree of consensus. Therefore, low agreement between experts when judging the quality of outputs for a particular model translates to bad model quality. The scores of agreement between evaluators are then used to rank the quality of the classifier models. We then check whether the ranking of models by the Wikipedia-based metric is aligned with the ranking of models by this manual evaluation.

[9]We used the 5-fold cross validation procedure in our experiment, however, we only evaluate the outputs links from one fold only in the manual evaluation.

1) EVALUATOR-CLASSIFIER RATIO

In this study, we evaluate the ability of the Wikipedia-based metrics using an ad-hoc "evaluator–classifier ratio" (*ECR*). Given an evaluator's judgment of a particular model's outputs, this metric divides the evaluator's positive opinions, i.e., when the evaluator deems the model's outputs as valid, by the total questions given in the manual evaluation. The result shows a relatively high agreement between each evaluator-classifier pair. The average *ECR* scores from the four judges are as follows: (1) GNB achieves a score of 0.904 for non-hidden links and 0.792 for hidden links, (2) SVM achieves a score of 0.659 for non-hidden links and 0.900 for hidden links, (3) MLP_10 achieves a score of 0.871 for non-hidden links and 0.842 for hidden links. This indicates that Wikipedia-based metrics are relatively reliable in verifying valid links.

2) INTER-ANNOTATOR AGREEMENT SCORES

To evaluate the reliability of the human assessment itself, we measure several inter-annotator agreement metrics. Many studies typically used the family of Kappa metrics, such as Cohen's Kappa [36] or Fleiss' Kappa [37]. Krippendorff's alpha [38], which is conceptually similar to the Kappa metrics (in using chance-corrected agreement), is also often used.

However, the Kappa metrics suffer from the Kappa paradoxes [39], [40], where the imbalanced class distribution leads to a low Kappa score (despite the highly observed agreement in reality). In our case, our evaluators frequently judge the models' outputs as valid and few "not valid" judgment–refer to Section VI-D1. This situation creates imbalanced marginals when computing the Kappa statistics. In simpler terms, the Kappa scores can be low even when the ECR score is high, in the case of imbalanced data. As the result, Kappa statistics may fail to represent the actual level of agreement of our evaluators.

To counter this issue, we use alternative (but less popular) agreement coefficients introduced by Kilem L. Gwet [41], called AC_1 and AC_2. The first metric, AC_1, is similar to Cohen's Kappa and calculates the agreement between two annotators. It is calculated as shown in Equation 7, where P_a represents the ratio of overall agreement and $P_{e\gamma}$ represents the chance agreement probability. In the case of multiple

TABLE 12. Samples of new topics for each of the direct approach's models and their respective CSO supertopics. The "new topic" column denotes completely new topics obtained from computer science papers that do not exist in the original CSO. The "CSO Topic" column contains topics that exist within the original CSO. The "description" column explains the relationship between the new topics and their CSO supertopics. In this study we only consider the "parent_of" (for the cso_topic – new_topic relationship) or "child_of" (for the new_topic – cso_topic relationship, i.e., the inverse of "parent of" relationship) relationships.

Model	CSO Topic	Description	New Topic
GNB	recommender systems	parent_of	context-aware recommendation
	computer vision	parent_of	visual recognition
	hashing	parent_of	hash code
	inference	parent_of	casual inference
	neural networks	parent_of	neural architecture search
	neural networks	parent_of	representation learning
	recommendation	parent_of	link recommendation
	classification methods	parent_of	supervised learning
SVM	machine learning	parent_of	metric learning
	learning	parent_of	learning-to-rank
	artificial intelligence	parent_of	educational robot
	hardware	parent_of	swap operation
	cycle	parent_of	graph cycle
	university	parent_of	academic performance
	neural network	parent_of	pre-training
	software	parent_of	text linking
MLP_10	social network	parent_of	link recommendation
	learning	parent_of	meta learning
	collaborative filtering	parent_of	item recommendation
	learning	parent_of	supervised learning
	robots	parent_of	educational robot
	social media	parent_of	social recommendation
	bayesian	parent_of	deep bayesian learning
	recommender system	parent_of	item-item product
MLP_50	search engine	parent_of	learning to rank
	learning	parent_of	neural embeddings
	neural network	parent_of	neural architecture search
	user interface	parent_of	usability
	learning	parent_of	representation learning
	learning	parent_of	meta learning
	neural networks	parent_of	supervised learning
	artificial intelligence	parent_of	educational robot
MLP_100	hashing	parent_of	hash code
	mobile	parent_of	empirical study
	learning	parent_of	conterfactual learning
	interaction design	parent_of	usability
	optimization	parent_of	neural architecture search
	learning	parent_of	dual-view sequential learning
	robots	parent_of	museum robot
	data mining	parent_of	social recommendation
MLP_200	machine learning algorithms	parent_of	meta learning
	machine learning	parent_of	metric learning
	support vector	parent_of	supervised learning
	recommendation	parent_of	skill recommendation
	robots	parent_of	educational robot
	bayesian	parent_of	deep bayesian learning
	neural network	parent_of	neural factorization machines
	robots	parent_of	community hri

annotators, we can use the AC_2 score, which was inspired by Krippendorff's alpha [38]. Equation 8 shows how to calculate the AC_2, where here P_a is the weighted overall agreement and P_e is the weighted chance agreement. We consider both types of agreements here: (1) agreement among all evaluators and (2) agreement between two evaluators.

$$AC_1 = \frac{P_a - P_{e\gamma}}{1 - P_{e\gamma}} \tag{7}$$

For the case of multiple (more than two) evaluators, AC_2 score (8), inspired by Krippendorff's alpha, can be used. This formula is very similar to Krippendorff's alpha with only the different definitions of P_a and P_e.

$$AC_2 = \frac{P_a - P_e}{1 - P_e} \tag{8}$$

Gwet's coefficients are especially suitable for imbalanced data, e.g., 85% to 95% agreement for "valid" category while

TABLE 13. The 5 × 2cv paired *t*-test scores of GNB, SVM and MLP_10 ($\alpha = 5\%$).

Model Pairs	t-statistics	p-value	Accept/Reject Null Hypothesis
GNB and SVM	-40.622	1.700×10^{-7}	Reject, the difference in performance is real
GNB and MLP_10	-34.895	3.600×10^{-7}	Reject, the difference in performance is real
SVM and GNB	40.622	1.700×10^{-7}	Reject, the difference in performance is real
SVM and MLP_10	9.620	2.058×10^{-5}	Reject, the difference in performance is real
MLP_10 and GNB	49.054	7.000×10^{-8}	Reject, the difference in performance is real
MLP_10 and SVM	-16.892	1.330×10^{-5}	Reject, the difference in performance is real

TABLE 14. AC_1 scores of the evaluator-pairs for the resulting non-hidden and hidden links from the direct approach's three best models. Best scores are shown in boldface.

Model	Evaluator Pairs											
	1 & 2		1 & 3		1 & 4		2 & 3		2 & 4		3 & 4	
	Non-hidden	*Hidden*	*Non-hidden*	*Hidden*	*Non-hidden*	*Hidden*	*Non-hidden*	*Hidden*	*Non-hidden*	*Hidden*	*Non-hidden*	*Hidden*
GNB	**0.731**	0.597	**0.756**	0.546	**0.642**	**0.627**	**0.909**	0.723	**0.820**	0.608	**0.843**	0.562
SVM	0.121	**0.706**	0.077	**0.706**	0.204	0.608	0.642	**1.000**	0.233	**0.803**	0.364	**0.803**
MLP_10	0.568	0.481	0.627	0.510	0.478	0.520	0.890	0.848	0.890	0.790	0.837	**0.803**

only 5% to 15% agreement for the 'non valid'' category. They are also capable in handling categorical, ordinal, interval, and missing data.

In Tables 14 and 15, it is shown the AC_1 and AC_2 scores are relatively high (more than 60% of agreement, with the best score of 100% agreement). Looking back at the results of the *ECR* scores explained in Section VI-D1, both AC_1 and AC_2 scores can therefore represent our evaluators' opinions more appropriately.

TABLE 15. AC_2 scores based on the opinion of four evaluators on the resulting links from the direct approaches three best models. Best scores are shown in boldface.

Model	Non-hidden	Hidden
GNB	**0.788**	0.610
SVM	0.263	**0.783**
MLP_10	0.731	0.674

E. DISCUSSION

In this section, we discuss the advantages and disadvantages of both the Indirect and Direct approaches. Furthermore, we explain the possible causes as to why some models are performing unsatisfactorily.

1) INDIRECT APPROACH

- **Low $F1$ Scores on Flat Classification**
  We identified factors that may have affected the final $F1$ scores (see Section VI-A3) of the flat classification models, as follows:
  1) **Overwhelming number of classification labels**, which is the case for both the Indirect and Direct approaches. For the Indirect approach specifically, there are 4,241 potential parent topics at most (''no_limit'').
  2) The data used in this study is **imbalanced**. As has been explained in Section IV-A1, one parent node may have many descendants, but others may only have a few.
- **Positive and Negative Affects of Minimizing Classification Targets**
  There are a total of 4,241 potential parents for the flat classification models. However, the number decreases drastically when we set the level limitation rule = 1, to 20 (''limit1_19'') or 31 (''limit1_30'') potential parents only. We observed that the performance of the classification models improved when we decreased the number of potential parents. Nevertheless, it should always be remembered that the more the limitation rule is applied, the more coarse-grained (or more general) the classification results are. This is because the classifier would focus more on the high-level (close to the root) parents when they are classifying new topics. Therefore, a higher performance score when using the level limitation rule does not translate to the improvement in performance in a real-life setting. This is also one of the reason why our Direct approach is more superior to the Indirect approach, as the classification models in the Direct approach can still produce meaningful links even under the ''no_limit'' condition.

2) DIRECT APPROACH

- **Low $F1_w$ Scores**
  Different from the reasons explained in Section VI-E1, the reason for the Direct approach's low Wikipedia-based $F1_w$ score is because most of the resulting links are not valid links. Recall that the $F1_w$ score measures the ability of a classifier to pick quality links within the Topic Graph during the filtering process. Since the idea of valid links is the final Topic Graph links that also appear in the Wikipedia Graph, we suspect that the current classifiers do not work well enough as link filters. The classifiers still pick some noisy links and consider them as important links, while in reality, the classifiers should be able to differentiate noisy links and important links within the noisy Topic Graph.
- **Classifier Works Better as a Filter**
  According to the resulting links shown in Tables 8 and 9, despite the low $F1_w$ scores on average, the classifiers still work better as filters than a *hidden link* predictor. It can be seen from the aforementioned tables that the resulting valid non-hidden links are higher in number compared to the valid hidden links.

- **More Hidden Links**
  Compared to the number of valid hidden links produced by the Indirect approach, the Direct approach can produce more valid hidden links by a large margin. This can be seen in Table 9 where all models of the Indirect approach produce 0 valid hidden links on average. Based on this result, our Direct approach is more suitable for automatically extending an ontology.
- **Relationship Between Filter Degree and $F1_w$ Score**
  The *filter degree* score can be used as supporting information of the model's capability by showing the level of its intelligence when applied to the specific link-filtering task. In particular, a combination of moderate *filter degree* score (30% to 70%) and large $F1_w$ score indicates a reliable classification model. This is because the model is able to filter out most of the unnecessary links and leave behind only the important links.
  We show an example of the connection between these metrics based on our previous experiment in [20] (full results are purposely not shown in this paper due to the difference in the experimental setting). Table 16 shows the *filter degree* and $F1_w$ scores obtained from our previous experiment [20].
  At the time, our best model was GNB while the worst model was *Random Forest* (RF). Table 16 shows that the GNB model is more reliable compared to the RF model. We can see that the RF model performs poorly in $F1_w$ score and its *filter degree* score is also extremely high. In contrast, the GNB model, which has a moderate (30%–70%) level of the *filter degree* score achieves a relatively stable $F1_w$ score across different batches.
  Despite numerous metrics we proposed in this paper, we argue that the number of total resulting links and our Wikipedia-based $F1_w$ should be the main metrics for automatic ontology extension evaluation. They are the easiest ones to interpret, while it is required for one to have an additional hands-on experience to understand the *filter degree*. Therefore, our recommended procedure (when using our metrics) is to judge the best model via the number of total links and $F1_w$ measure first, and then use the *filter degree* score as a supporting metric for an additional analysis. In other words, one metric would not be sufficient to describe all phenomena (as have been demonstrated in this paper). Obtaining the highest performance in one particular metric may not constitute a state-of-the-art performance. We recommend one to analyse comprehensively using several metrics.
- **Human Evaluation for Reliability**
  We then evaluate the top-performing classifiers from the Direct approach using an additional human evaluation. We measure the agreement (1) between classification model to each of the human evaluators and (2) between (all and pairs of) human evaluators. In the former case, we used *ECR* to see what percentage of the resulting links by the classification models were agreed upon by each of the evaluators. The scores show that on average, the evaluators have relatively high agreement (around 66% to 90%) with each of the classification models.
  After confirming the *ECR* score, we further investigate the inter-evaluator agreement (via Gwet's AC_1 and AC_2 inter-annotator agreement scoring). The same models achieve moderate to high-agreement scores between evaluators. We argue that this indicates the reliability of our model. The GNB model, in particular, achieves the highest agreement scores for non-hidden links. On the other hand, the SVM model achieves the highest agreement for hidden links. But looking at the overall picture, GNB is still the best model as it achieves good scores for both non-hidden and hidden links, while the SVM model only performed well for hidden links.

TABLE 16. ***Filter degree*** **and** $F1_W$ **score combination of** ***Gaussian Naive Bayes*** **and** ***Random Forest*** **from our older experiment [20]. Here the "batch" column represents data different batches (different numbers of paper abstracts from which the new topics are obtained).**

Batch	Gaussian Naive Bayes		Random Forest	
	filter degree	$F1_w$	*filter degree*	$F1_w$
1	0.702	0.274	0.979	0.044
2	0.529	0.286	0.980	0.056
3	0.390	0.296	0.952	0.141
4	0.335	0.269	0.968	0.099

F. OBSERVATION

In our experiment, our Direct approach's *Gaussian Naive Bayes* showed the highest potential for realizing a fully automated ontology extension. This model could produce reliable non-hidden links, however, the *Support Vector Machine* could produce reliable hidden links. We consider combining these two models to improve the performance, in the future.

For the moment, the Direct approach's *Gaussian Naive Bayes* produced the highest number of valid hidden links, and therefore, this classifier is more reliable in terms of production compared to the *Support Vector Machine* model. Aside from these two classification models, the remaining *Multi Layer Perceptron* model also showed promising performance and agreement between evaluators. The confidence in selecting the Direct approach's classification models also comes from their stable ($F1_w$) scores and total produced links (See Section VI-C) where *Gaussian Naive Bayes* is always (one of) the best performing model(s) with *Support Vector Machine* always following closely behind.

In the case of the Indirect approach, from only looking at the number of resulting new links of its classification model, the *Gaussian Naive Bayes* model is the most reliable. The other models tend to result in few new links and topics.

Additionally, most of the performance scores of the Indirect approach's classification models are unsatisfactory and inconsistent. Despite the little potential of the Indirect approach's *Gaussian Naive Bayes* model, the Indirect approach itself is still too risky to be called reliable due to their low $F1_w$ scores. Table 5 shows the comparison of $F1_w$ scores between the Indirect and Direct approaches.

1) ACHIEVED RESEARCH GOALS

In this paper, we achieved the following research goals:

1) Extending the original CSO with as many new up-to-date Computer Science topics as possible, obtained from newly published papers from 2018 to 2020.
2) Enriching the original CSO with new relationships between topics.
3) Obtaining an extended CSO which will be useful for (but not limited to) our future scientific paper on a recommender system project.

2) ANSWERED RESEARCH QUESTIONS

1) **Is an automated ontology extension approach feasible?** Depending on the purpose of the ontology extension, an automated ontology extension can be either reliable or unreliable. Since we plan to use the extended ontology for a scientific paper recommender, the proposed automatic ontology extension approach is relatively acceptable for this purpose. This is because the resulting extended ontology consists of the enriched information (i.e., new Computer Science Topics and their links) and can aid the recommender system in recommending related items to users.
   On the other hand, the manual link evaluation presented in this paper shows that our human experts tend to have high agreement with the resulting links produced by the top classifiers (see Section VI-D1), as well as high agreement with each other (see Section VI-D2). Therefore, our automatic process might be good enough for extending an ontology in general. However, as explained above, not all tasks are as flexible as the recommendation task. There are tasks requiring high-accuracy. When a high level of accuracy is required, it is better to evaluate the extended ontology using a more reliable ground truth (i.e., even a more reliable source of knowledge than Wikipedia)
2) **Which approach, Direct or Indirect, is better at recognising new topics and properly linking them into the CSO ontology?** The Direct approach is more preferable than the Indirect approach at extending ontology. The best performing classification model for the Direct approach is the *Gaussian Naive Bayes* algorithm according to numerous evaluation aspects.

VII. CONCLUSION AND FUTURE WORKS

This paper presented work on the automatic ontology extension task for the Computer Science Ontology (CSO). We proposed a novel ''Direct'' approach and compared its performance to the existing ''Indirect'' approach. Both approaches are implemented using various classification models.

The main difference between both approaches is whether there is a *direct involvement* of news topics during the training process. For the Indirect approach, a topic classifier is first trained on the original CSO, which is then used to classify new topics. On the other hand, the Direct approach first performs an initial extension of the original CSO with new topics, and then a classifier filters out noisy links as well as to predict potential valuable hidden links in the initial extended ontology.

We evaluated the classification models (under both approaches) using various metrics: Wikipedia-based evaluation metrics, *filter degree*, and the number of valid predicted links. Our empirical evaluation was also followed by a statistical significance test and a manual evaluation by four experts. Our experiment showed that the Direct approach has an overall better and more reliable performance out of the two approaches on all the aforementioned evaluation aspects.

Based on several evaluation aspects, we conclude the best model for implementing the Direct approach as follows (ordered by the overall performance): *Gaussian Naive Bayes*, *Support Vector Machine*, and *Multilayer Perceptron* (with 10 hidden layer neurons). We then used the *Gaussian Naive Bayes* to extend the CSO with both its predicted non-hidden and hidden links. This is because the *Gaussian Naive Bayes* model could produce the highest number of total valid links and has a relatively high agreement to human experts when compared to other models in this aspect. As the result, only from using the test data, we can increase the CSO with approximately 1,140 new topics (around an 8.14% increase compared to the original version) and 1,716 new links.

This study verified that automatic ontology extension is feasible, at least for the Computer Science domain. We will explore various methods to improve our Direct approach and further enrich the Computer Science Ontology. We also plan to investigate the use of hierarchical classification techniques under our Direct approach. Last but not least, we will evaluate the reliability of an automatically extended ontology by showing the difference of performance when applying our extended CSO versus the original one in the downstream task of a recommendation system.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers and Jan Wira Gotama Putra (Tokyo Institute of Technology) for their helpful feedback.

REFERENCES

[1] A. Konys, ''Knowledge repository of ontology learning tools from text,'' *Proc. Comput. Sci.*, vol. 159, pp. 1614–1628, Jan. 2019.
[2] P. Cimiano and J. Völker, ''Text2onto—A framework for ontology learning and data-driven change discovery,'' in *Proc. 10th Int. Conf. Appl. Natural Lang. Inf. Syst. (NLDB)*, in Lecture Notes in Computer Science, vol. 3513, A. Montoyo, R. Munoz, and E. Metais, Eds. Alicante, Spain: Springer, 2005, pp. 227–238.
[3] M. T. Pazienza and A. Stellato, ''The protégé ontoling plugin-linguistic enrichment of ontologies in the semantic web,'' in *Proc. 4th Int. Semantic Web Conf. (ISWC)*, Nov. 2005, pp. 1–2.
[4] J. Shen, Z. Wu, D. Lei, C. Zhang, X. Ren, M. T. Vanni, B. M. Sadler, and J. Han, ''HiExpan: Task-guided taxonomy construction by hierarchical tree expansion,'' in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2180–2189.
[5] A. A. Salatino, T. Thanapalasingam, A. Mannocci, F. Osborne, and E. Motta, ''The computer science ontology: A large-scale taxonomy of research areas,'' in *The Semantic Web—ISCW 2018*, vol. 1137, D. Vrandečić *et al.*, Eds. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-030-00668-6_12.
[6] D. Man, ''Ontologies in computer science,'' in *Didactica Mathematica*, vol. 31. Cluj-Napoca, România: Babeş-Bolyai Univ., 2013, pp. 43–46. [Online]. Available: http://www.math.ubbcluj.ro/~didactica/pdfs/2013/index.html

[7] C. Zhang, F. Tao, X. Chen, J. Shen, M. Jiang, B. Sadler, M. Vanni, and J. Han, "TaxoGen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 2701–2709.

[8] H. H. Alrehamy and C. Walker, "SemCluster: Unsupervised automatic keyphrase extraction using affinity propagation," in *Advances in Intelligent Systems and Computing*, vol. 650, F. Chao, S. Schockaert, and Q. Zhang, Eds. Cham, Switzerland: Springer, 2017, doi: 10.1007/978-3-319-66939-7_19.

[9] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 10, pp. 1825–1837, Oct. 2018.

[10] J. Liu, J. Shang, C. Wang, X. Ren, and J. Han, "Mining quality phrases from massive text corpora," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, May 2015, pp. 1729–1744.

[11] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, "YAKE! Keyword extraction from single documents using multiple local features," *Inf. Sci.*, vol. 509, pp. 257–289, Jan. 2020.

[12] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "A text feature based automatic keyword extraction method for single documents," in *Advances in Information Retrieval* (Lecture Notes in Computer Science), vol. 10772, G. Pasi, B. Piwowarski, L. Azzopardi, and A. Hanbury, Eds. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-76941-7_63.

[13] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "YAKE! Collection-independent automatic keyword extractor," in *Advances in Information Retrieval* (Lecture Notes in Computer Science), vol. 10772, G. Pasi, B. Piwowarski, L. Azzopardi, and A. Hanbury, Eds. Cham, Switzerland: Springer, 2018, doi: 10.1007/978-3-319-76941-7_80.

[14] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic keyword extraction from individual documents," *Text Mining, Appl. Theory*, vol. 1, pp. 1–20, Mar. 2010.

[15] A. Ayadi, A. Samet, F. D. B. de Beuvron, and C. Zanni-Merk, "Ontology population with deep learning-based NLP: A case study on the biomolecular network ontology," *Proc. Comput. Sci.*, vol. 159, pp. 572–581, Jan. 2019.

[16] W. Liu, A. Weichselbraun, A. Scharl, and E. Chang, "Semi-automatic ontology extension using spreading activation," *J. Universal Knowl. Manage.*, no. 1, pp. 50–58, 2005.

[17] I. Novalija and D. Mladenic, "Semi-automatic ontology extension using text mining," in *Proc. Conf. Data Mining Data Warehousing (SiKDD)*, 2009, pp. 1–4.

[18] I. Novalija and D. Mladenić, "Content and structure in the aspect of semi-automatic ontology extension," in *Proc. ITI 32nd Int. Conf. Inf. Technol. Interfaces*, Jun. 2010, pp. 115–120.

[19] S. Althubaiti, Ş. Kafkas, M. Abdelhakim, and R. Hoehndorf, "Combining lexical and context features for automatic ontology extension," *J. Biomed. Semantics*, vol. 11, no. 1, pp. 1–13, Dec. 2020.

[20] N. C. Santosa, J. Miyazaki, and H. Han, "Flat vs. hierarchical: Classification approach for automatic ontology extension," in *Proc. 13th Forum Data Eng. Inf. Manage.*, Jul. 2021, pp. 1–10.

[21] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[22] N. N. Daud, S. H. A. Hamid, M. Saadoon, F. Sahran, and N. B. Anuar, "Applications of link prediction in social networks: A review," *J. Netw. Comput. Appl.*, vol. 166, Sep. 2020, Art. no. 102716.

[23] P. Wang, B. Xu, Y. Wu, and X. Zhou, "Link prediction in social networks: The state-of-the-art," *Sci. China Inf. Sci.*, vol. 58, no. 1, pp. 1–38, Jan. 2015.

[24] T. Murata and S. Moriyasu, "Link prediction of social networks based on weighted proximity measures," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, Nov. 2007, pp. 85–88.

[25] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos, "Fast and accurate link prediction in social networking systems," *J. Syst. Softw.*, vol. 85, no. 9, pp. 2119–2132, 2012.

[26] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, and Y. Elovici, "Link prediction in social networks using computationally efficient topological features," in *Proc. IEEE 3rd Int. Conf. Privacy, Secur., Risk Trust IEEE 3rd Int. Conf. Social Comput.*, Oct. 2011, pp. 73–80.

[27] A. Boyd, "Using Wikipedia edits in low resource grammatical error correction," in *Proc. EMNLP Workshop W-NUT: 4th Workshop Noisy User-Generated Text*, 2018, pp. 79–84.

[28] F. Chiarello, L. Trivelli, A. Bonaccorsi, and G. Fantoni, "Extracting and mapping industry 4.0 technologies using Wikipedia," *Comput. Ind.*, vol. 100, pp. 244–257, Sep. 2018.

[29] J. Kim, S. Kim, and C. Lee, "Anticipating technological convergence: Link prediction using Wikipedia hyperlinks," *Technovation*, vol. 79, pp. 25–34, Jan. 2019.

[30] C.-C. Ni, K. Sum Liu, and N. Torzec, "Layered graph embedding for entity recommendation using Wikipedia in the Yahoo! Knowledge graph," in *Proc. Companion Proc. Web Conf.*, Apr. 2020, pp. 811–818.

[31] H. K. Azad and A. Deepak, "A new approach for query expansion using Wikipedia and WordNet," *Inf. Sci.*, vol. 492, pp. 147–163, Aug. 2019.

[32] F. Osborne and E. Motta, "Klink-2: Integrating multiple web sources to generate semantic topic networks," in *Proc. 14th Int. Semantic Web Conf. (ISWC)*, in Lecture Notes in Computer Science, vol. 9366, M. Arenas *et al.*, Eds. Cham, Switzerland: Springer, Oct. 2015, doi: 10.1007/978-3-319-25007-6_24.

[33] F. Osborne, E. Motta, and P. Mulholland, "Exploring scholarly data with rexplore," in *The Semantic Web—ISWC 2013*, in Lecture Notes in Computer Science, vol. 8218, H. Alani *et al.*, Eds. Berlin, Germany: Springer, 2013, doi: 10.1007/978-3-642-41335-3_29.

[34] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[35] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, 1998.

[36] J. Cohen, "A coefficient of agreement for nominal scales," *Educ. Psychol. Meas.*, vol. 20, no. 1, pp. 37–46, 1960.

[37] J. L. Fleiss, "Measuring nominal scale agreement among many raters," *Psychol. Bull.*, vol. 76, no. 5, p. 378, 1971.

[38] K. Krippendorff, "Computing Krippendorff's alpha-reliability," Annenberg School Commun., Univ. Pennsylvania, Philadelphia, PA, USA, 2011. [Online]. Available: https://repository.upenn.edu/asc_papers/43/

[39] A. R. Feinstein and D. V. Cicchetti, "High agreement but low kappa: I. The problems of two paradoxes," *J. Clin. Epidemiology*, vol. 43, no. 6, pp. 543–549, Jan. 1990.

[40] D. V. Cicchetti and A. R. Feinstein, "High agreement but low kappa: II. Resolving the paradoxes," *J. Clin. Epidemiol.*, vol. 43, no. 6, pp. 551–558, Jan. 1990.

[41] K. L. Gwet, "Computing inter-rater reliability and its variance in the presence of high agreement," *Brit. J. Math. Statist. Psychol.*, vol. 61, no. 1, pp. 29–48, 2008.

NATASHA C. SANTOSA received the B.Sc. degree in computer science from Gadjah Mada University, Indonesia, and the M.Eng. degree in artificial intelligence from the Tokyo Institute of Technology, Japan, where she is currently pursuing the Ph.D. degree with the Computer Science Department. Her research interests include recommender systems, natural language processing, data mining, and machine learning.

JUN MIYAZAKI (Member, IEEE) received the B.E. degree from the Tokyo Institute of Technology (Tokyo Tech.), Tokyo, Japan, in 1992, and the M.S. and Ph.D. degrees from the Japan Advanced Institute of Science and Technology (JAIST), Japan, in 1994 and 1997, respectively. He joined Tokyo Tech., in 2013, where he is currently a Professor with the School of Computing and the Deputy Director of the Global Scientific Information and Computing Center. Before joining Tokyo Tech., he worked with the Nara Institute of Science and Technology (NAIST) as an Associate Professor. His research interests include database systems, cloud computing, and information recommendation. He is a member of ACM, IEICE, IPSJ, and DBSJ.

HYOIL HAN received the Ph.D. degree in computer science and engineering from the University of Texas at Arlington, in 2002. She is currently an Associate Professor with the School of Information Technology, Illinois State University, USA. She worked at Samsung Electronics and Korea Telecom, before obtaining her Ph.D. degree. Her research interests include machine learning, natural language processing, big data management, and applying AI to security.