Faceted Hierarchy: A New Graph Type to Organize Scientific Concepts and a Construction Method

Qingkai Zeng[†], **Mengxia Yu**[†], **Wenhao Yu**[†], **Jinjun Xiong**[‡], **Yiyu Shi**[†], **Meng Jiang**[†]
[†]Department of Computer Science and Engineering,
University of Notre Dame, Notre Dame, Indiana, USA
[‡]IBM T. J. Watson Research Center, Yorktown Heights, NY USA
[†]{qzeng, myu2, wyu1, yshi4, mjiang2}@nd.edu,[‡]jinjun@us.ibm.com

Abstract

On a scientific concept hierarchy, a parent concept may have a few attributes, each of which has multiple values being a group of child concepts. We call these attributes *facets*: classification has a few facets such as *application* (e.g., face recognition), *model* (e.g., svm, knn), and *metric* (e.g., precision). In this work, we aim at building *faceted concept hierarchies* from scientific literature. Hierarchy construction methods heavily rely on hypernym detection, however, the faceted relations are *parent-to-child* links but the hypernym relation is a multi-hop, i.e., *ancestor-to-descendent* link with a specific facet "type-of". We use information extraction techniques to find synonyms, sibling concepts, and ancestor-descendent relations from a data science corpus. And we propose a hierarchy growth algorithm to infer the parent-child links from the three types of relationships. It resolves conflicts by maintaining the acyclic structure of a hierarchy.

1 Introduction

Concept hierarchies play an important role in knowledge discovery from scientific literature. Concepts are expected to be organized in a hierarchical structure like chapters-to-sections-to-subsections in textbooks. In this work, we propose a new representation of scientific concept hierarchy, called *faceted concept hierarchy*. Under this hierarchy, the links should not only carry parent-to-child relations but also the semantic relations (*facets*) between the concepts. Figure 1 presents a part of the faceted hierarchy in Data Science. The parent node is "classification" and the child concepts of it are excepted to be grouped into three facets, each of which has three child-concepts. One example of the *faceted relation* we define is as follows:

parent("classification", "svm"): "models",

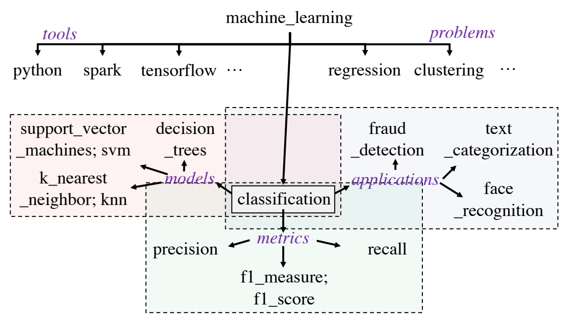


Figure 1: The idea of faceted concept hierarchy from Data Science publications: For student learning, concepts are expected to be organized in a hierarchical structure. For example, here the nine child-concepts of "classification" (in dashed line blocks) should be grouped into three facets ("models", "applications", and "metrics").

which is more complete than "type-of" relations in the works that focused on taxonomy or ontology induction (Liang et al., 2017; Gupta et al., 2017; Zhang et al., 2018; Liu et al., 2018) like this:

type-of("svm", "classification model").

The basic units of the hierarchy include concept nodes and their parent-to-child relations. Three types of essential structural relations are expressed in paper texts and can be used to infer the parent-to-child relations. The relation types include (1) synonym (concept names on the same node), (2) sibling (concept nodes having the same parent), and (3) ancestor-to-descendant (nodes on the direct descending line). The task of hierarchy construction has three challenges. First, there is no sufficient human annotated data or available distant supervisory sources to feed into (deep) learning models. It is necessary to extract the concepts and relations in an *unsupervised* manner. Second, the extracted relations could be noisy at the long tail of the frequency distribution. When inferring the parent-to-child relations, the algorithm should consider the trustworthiness of the synonym, sibling, and ancestor relations. Also, it is important to detect redundant or conflicting relations (links) on the hierarchy. Third, it requires a joint process of clustering child-concepts into the parent concept's facets and identifying words as facet indicators.

We propose a novel framework *HiGrowth* that grows faceted hierarchies from literature data. The framework has five modules: (**M1**) scientific concept extraction, (**M2**) concept typing, (**M3**) hierarchical relation extraction, (**M4**) hierarchy growth, and (**M5**) facet discovery. The M1–M3 NLP modules were implemented in an unsupervised manner. First, we use two complementary keyphrase mining tools to extract concepts: one is rule-based and the other is a statistical learning method. Second, we use a KNN-based method, simple and effective, to assign types (e.g., $*Problem*, $*Method*) to the concepts. Third, we use textual patterns to extract the hierarchical relations (i.e., synonym, sibling, and ancestor). To address the second challenge, we design an efficient algorithm that grows a concept hierarchy by scanning the set of relation tuples (sorted by their frequency from the highest to the lowest) just once and inferring parent-to-child relations. This algorithm will be able to identify unnecessary, invalid, and redundant links during the process of hierarchy growth in spite of serious noise at the long tail. Finally, we use a word clustering method to discover the facets of every parent concept and assign child concepts to each of the facets.

Thirty-two junior/senior students who took the Data Science course in Spring 2018 were asked to manually label the parent-child concept pairs. We finalize a set as ground-truth if the pair was labelled by more than half of the participants. The F1 score of building the parent-to-child links is 0.73. The F1 score of 2-hop paths is 0.69. Both precision values are above 0.99, showing that the links in the hierarchy are precise because of the careful design of the growth algorithm, but the pattern-based methods have limitations of finding all possible relations.

2 *HiGrowth* – Part I: Information Extraction Components (M1 – M3)

2.1 Data Description

We collected *full text*, all sections including abstract, introduction, and experiments, of 5,850 papers in the proceedings of ACM SIGKDD 1994–2015, IEEE ICDM 2001–2015, The Web Conference 2001–2015 and ACM WSDM 2008–2015.

Table 1: Neighboring words for concept typing.

Type	Triggers on the left	Triggers on the right
$*Problem*	problem, problems	task, tasks, demands
$*Method*	methods	method, model, algorithm
$*Object*	number	function
$*Metric*	measure, terms	measures, scores, values

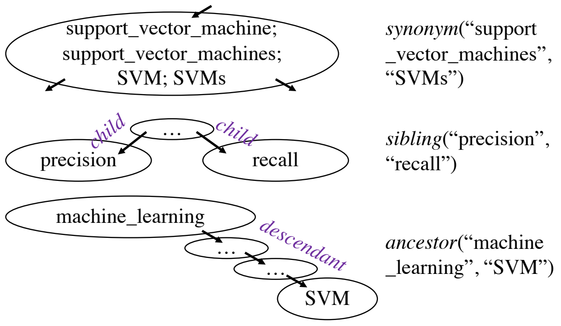

support_vector_machine; support_vector_machines; SVM; SVMs
synonym("support _vector_machines", "SVMs")
precision
child
...
child
recall
sibling("precision", "recall")
machine_learning
...
descendant
...
SVM
ancestor("machine _learning", "SVM")

Figure 2: Three types of hierarchical relations.

2.2 M1: Scientific Concept Extraction

We use phrase mining tools, *AutoPhrase* (Shang et al., 2018) & *SCHBase* (Adar and Datta, 2015), to extract scientific concepts from data science papers. *AutoPhrase* adopted distant supervision and large-scale statistical techniques; *SCHBase* focused on a tendency to expand keyphrases by adding terms, coupled with a pressure to abbreviate to retain succinctness in academic writing.

2.3 M2: Concept Typing

We use a simple but effective method to classify the concepts into four types: $*Problem* (e.g., "fraud_detection"), $*Method* (e.g., "svm"), $*Object* (e.g., "frequent_patterns"), and $*Metric* (e.g., "accuracy"). We assume that the neighboring non-stop word indicates the concept's type, for example, the trigger word "problem" in the sentence "...the problem of fraud detection" suggests that "fraud_detection" is a $*Problem*. We manually select a set of 20 trigger words that indicate concept types when they appear left/right next to the concepts. Table 1 shows a few examples. If in the text one concept has a left/right neighboring word in the set, the corresponding type gets one vote. For each concept, we count the votes on every type and use the strategy of majority voting (MajVot) to determine the predicted type (i.e., the most voted).

2.4 M3: Hierarchical Relation Extraction

In order to find the relations in an unsupervised manner on the scientific text, we use textual patterns, mainly Hearst Patterns (Hearst, 1992), to accurately extract three types of hierarchical relations, where X and Y are two concept names:

- *synonym*(X, Y), if X and Y will be included in the same concept node on the hierarchy;
- *sibling*(X, Y), if the concept nodes of X and Y will have the parent node;
- *ancestor*(X, Y), if there will be a path from the concept node of X to the node of Y.

Note that *synonym* and *sibling* relations are symmetric, while *ancestor-to-descendant* is asymmetric (see Figure 2).

Find *synonym*(X, Y). Two ideas to find synonym concepts: First, the plural form of a noun or noun-phrase concept can be considered as a synonym, for example, we have *synonym*("SVM", "SVMs") and *synonym*("decision_tree", "decision_trees"). Second, the abbreviation inside of parentheses can be considered as a synonym of the full name preceding the parenthesis. We have *synonym*("support_vector_machines", "SVMs") from text "...Support Vector Machines (SVMs)...".

Find *ancestor*(X, Y). Hearst patterns such as

- X such as $\{Y_1, Y_2, \ldots,$ (and|or)$\}$ Y_n,
- X $\{,\}$ including $\{Y_1, Y_2, \ldots,$ (and|or)$\}$ Y_n,

have been often used to find "isA" relation or called hypernym for taxonomy construction: Y_i (e.g., "dog") is a kind of X (e.g., "mammal"). However, we expect to extract *faceted hierarchical relations* such as

- *ancestor*("machine_learning", "SVM"): models;
- *ancestor*("machine_learning", "classification"): tasks;
- *ancestor*("classification", "SVM"): models;

instead of

- *isA*("machine_learning_models", "SVM");
- *isA*("machine_learning_tasks", "classification");
- *isA*("classification_models", "SVM"),

if the text contains

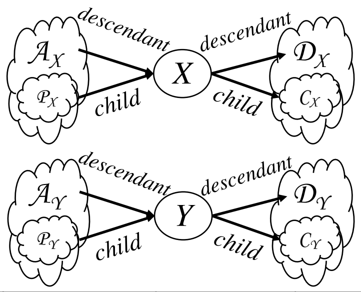
- ...machine learning models such as SVM...;
- ...machine learning tasks such as classification...;
- ...classification models such as SVM...,

especially when "machine_learning" has been extracted as a concept. Note that we are *not* confident to say every relation given by pattern matching is *parent-to-child*. We denote the relation as *ancestor*. We expect that "machine_learning" connects to "SVM" through "classification" on the hierarchy instead of a direct connection.

Therefore, we modify the patterns as below:

- X <pl> such as $\{Y_1, \ldots,$ (and|or)$\}$ Y_n,
- X <pl> including $\{Y_1, \ldots,$ (and|or)$\}$ Y_n,

where <pl> is the plural form of a noun or noun phrase, e.g., "models" and "tasks". We extract *ancestor*(X, Y_i) from the above patterns. We will



	synonym(X, Y)	*sibling(X, Y)*	*ancestor(X, Y)*
PASS (no need to add)	–	$\mathcal{P}_X \cap \mathcal{P}_Y \neq \emptyset$	$Y \in \mathcal{D}_X \Leftrightarrow X \in \mathcal{A}_Y$
STOP (should not add)	$\mathcal{P}_X \cap \mathcal{P}_Y \neq \emptyset$ OR $Y \in \mathcal{D}_X$ OR $X \in \mathcal{D}_Y$	$Y \in \mathcal{D}_X$ OR $X \in \mathcal{D}_Y$	$\mathcal{P}_X \cap \mathcal{P}_Y \neq \emptyset$ OR $\mathcal{C}_X \cap \mathcal{C}_Y \neq \emptyset$ OR $X \in \mathcal{D}_Y$
If matched none of the above conditions, then *UPDATE* the hierarchy $\mathcal{H}$ with the relation between X and Y.			

Figure 3: Check if the relation is unnecessary ("PASS") or invalid ("STOP") for updating $\mathcal{H}$.

infer concrete *parent-to-child* relations and parent concept's facets in the next section.

Find *sibling*(X, Y). Shorter patterns in which the ancestor concept names are missing occur more frequently in the text, for example:

- (<pl>) such as $\{Y_1, Y_2, \ldots,$ (and|or)$\}$ Y_n,
- (<pl>) including $\{Y_1, Y_2, \ldots,$ (and|or)$\}$ Y_n,

and other patterns like

- $Y_1, Y_2, \ldots,$ and|or (other) <pl>.

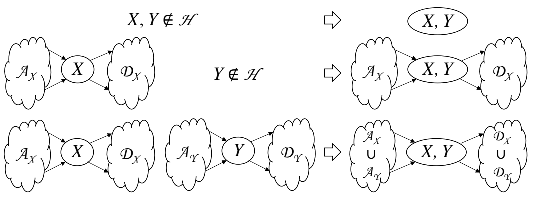
We extract *sibling*(Y_i, Y_j) from these patterns. The number of *sibling* relations is more than the number of the *ancestor* relations, and the *sibling* relations, e.g., *sibling*("precision", "recall"), bring useful information to hierarchy induction, say, Y_i and Y_j have the same *parent* concept node.

We use the strategy of majority voting to choose one relation type for each pair of concepts. We assume that a pair of concepts can have no or only one relation among *synonym*, *sibling*, and *ancestor*. However, the relational extractions may still be noisy due to the long tail. Next we discuss how to construct a high-quality concept hierarchy from a set of the three types of relations with noise.

3 *HiGrowth* – Part II: Hierarchy Growth (M4) and Facet Discovery (M5)

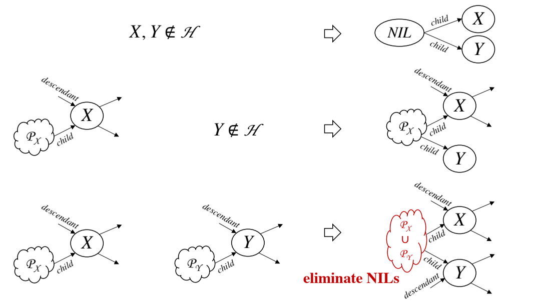
3.1 M4: The Hierarchy Growth Algorithm

Given a set of relations *rel*(X, Y) and their support (i.e., frequency), construct a hierarchy $\mathcal{H}$ in which the links are directional indicating *parent-*

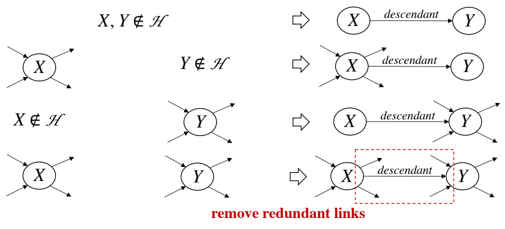


$X, Y \notin \mathcal{H}$
$Y \notin \mathcal{H}$
X, Y
$\mathcal{A}_X$
$\mathcal{D}_X$
$\mathcal{A}_Y$
$\mathcal{D}_Y$
$\mathcal{A}_X \cup \mathcal{A}_Y$
$\mathcal{D}_X \cup \mathcal{D}_Y$

(a) Grow $\mathcal{H}$ with symmetric *synonym*(X, Y).



(b) Grow $\mathcal{H}$ with symmetric *sibling*(X, Y).



(c) Grow $\mathcal{H}$ with asymmetric *ancestor*(X, Y).

Figure 4: Hierarchy growth with a new relation.

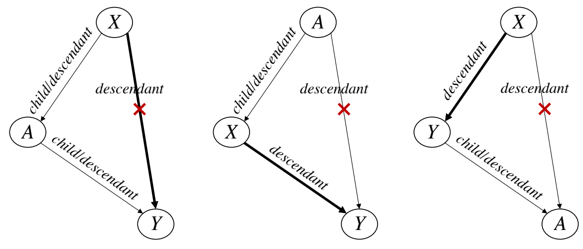
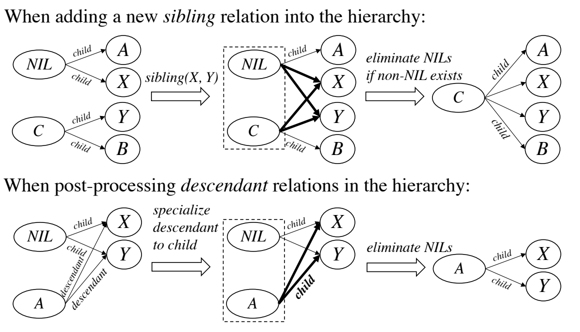


Figure 5: Remove redundancy when adding *ancestor*(X, Y) to the hierarchy.

to-child relations between concepts, where *rel* $\in$ $\{$*synonym*, *sibling*, *ancestor*$\}$. $\mathcal{H}$ should have no *unnamed* nodes, and have no *unnecessary* or *invalid* or *redundant* links. Specifically, the *unnecessary* means that the relation is correct but it does not contain extra information for the hierarchy. We will define these characteristics when we introduce each step of it in details. An overview of the algorithm comes as below.

- Initialize $\mathcal{H}$ as empty;
- Scan the relations from the highest support to the lowest:
  - Check if the relation is unnecessary or invalid to be added into $\mathcal{H}$ (see Figure 3). Skip it if yes.
  - Grow the hierarchy $\mathcal{H}$ with this relation (see Figure 4).
  - Remove redundant links when the relation is *ancestor* (see Figure 5).
- Narrow down *ancestor* relations to *parent-to-child* when the scan completes (see Figure 6).



When adding a new *sibling* relation into the hierarchy:
NIL
child
sibling(X, Y)
eliminate NILs if non-NIL exists
When post-processing *descendant* relations in the hierarchy:
specialize descendant to child
descendant
eliminate NILs

Figure 6: Two scenarios that NIL nodes can be eliminated when finalizing the hierarchy.

We denote different sets of connected nodes given a concept node X as below (see Figure 3):

- $\mathcal{P}_X$ is the set of parent nodes of X: there is at least one direct link from $\forall Z \in \mathcal{P}_X$ to X;
- $\mathcal{C}_X$ is the set of child nodes of X: there is at least one direct link from X to $\forall Z \in \mathcal{C}_X$;
- $\mathcal{A}_X$ is the set of ancestor nodes of X: there is at least one path but no direct link from $\forall Z \in \mathcal{A}_X$ to X;
- $\mathcal{D}_X$ is the set of descendant nodes of X: there is at least one path but no direct link from X to $\forall Z \in \mathcal{D}_X$.

Check if a relation is invalid (Figure 3). Given a new relation *synonym*(X, Y), if there has been any other relation between X and Y such as *ancestor* (i.e., $X \in \mathcal{D}_Y$ or $Y \in \mathcal{D}_X$) or *sibling* (i.e., $\mathcal{P}_X \cap \mathcal{P}_Y \neq \emptyset$), this new relation is invalid to be added to the $\mathcal{H}$. Given *sibling*(X, Y), if X and Y have at least one parent, we skip; if there has been an *ancestor* relation between X and Y, the *sibling* relation is invalid. Given *ancestor*(X, Y), if there has been path from X to Y (i.e., $Y \in \mathcal{D}_X$), we skip it; if there has been a *sibling* relation (i.e., $\mathcal{P}_X \cap \mathcal{P}_Y \neq \emptyset$) or a descendant relation (i.e., $X \in \mathcal{D}_Y$), the *ancestor* relation is invalid.

Grow the hierarchy $\mathcal{H}$ with a new relation (Figure 4). We sort valid relations by their frequencies. For *synonym*(X, Y), we merge node X and Y in $\mathcal{H}$: if neither was in $\mathcal{H}$, we create a new isolated node named "X, Y"; if one of them existed in $\mathcal{H}$, we update the name of the existing node as "X, Y"; if both existed, we merge their ancestor nodes as the new ancestor node $\mathcal{A}_X \cup \mathcal{A}_Y$, and we

Table 2: Data science concept examples extracted by two complementary phrase mining tools.

Keyword	Count	Keyphrase	Count
clustering	22,607	data_mining	8,120
classification	19,488	machine_learning	4,195
accuracy	18,108	feature_selection	3,320

(a) *AutoPhrase* finds quality keywords/phrases of good statistical features (e.g., frequency, concordance).

Keyword	Count	Keyphrase	Count
SVM	5,774	least_squares_support_vector_machine	4
LDA	3,548	root-mean-square_error	3
AUC	2,542	block_coordinate_gradient_descent	1

(b) Some typical examples of acronyms and abbreviation expansions found by *SCHBase*.

merge their descendant nodes as the new descendant node $\mathcal{D}_X \cup \mathcal{D}_Y$.

For *sibling*(X, Y), if neither of the concepts existed, we create a "NIL" node as the parent node to each concept node; if one of them existed, for each parent node in $\mathcal{P}_X$, we add Y as a child node of it; if both existed, we merge their parent nodes as the parent node of each and eliminate the NILs.

For *ancestor*(X, Y), we add a descendant link from X to Y. When X and Y are in $\mathcal{H}$, we eliminate the NILs and remove the redundant links.

When adding a new relation *sibling*(X, Y), we merge their parent nodes. If there has been at least one non-NIL node in the set of parent nodes, we remove the NILs. When adding an *ancestor* node of either X or Y, if they share a NIL parent node, we remove the NIL node.

Remove redundant links when growing with *ancestor*(X, Y) (Figure 5). On the concept hierarchy, we allow only one path from an ancestor node to a descendant node. Therefore, when we add a new *ancestor*(X, Y), there are three situations of having a redundant link. First, if there has been a path from X to Y, the new relation is redundant. For example, suppose on $\mathcal{H}$, A ("svm") is a descendant node of X ("classification") and Y ("ls-svm") is a descendant node of A ("svm"). Then a new relation *ancestor*("classification", "ls-svm") is actually inferable so it is redundant. We do not add it to the hierarchy. For the other two situations, we also remove the existing, redundant link in the hierarchy.

3.2 M5: Facet Word Discovery using Context Word Clustering

For each parent node, we decompose a 3-order tensor, *child node* by *type of child node* by *context word*, in which each entry is the count of the context word (e.g., "models") used in the patterns (e.g., "$*Problem*:classification models such as $*Method*:naïve_bayes and $*Method*:svm") that indicate the semantic relation between the parent node (e.g., "classification") and child node (e.g., "svm"). The decomposition assigns a cluster of context words to each group of child nodes. We consider the most frequent context word in the cluster as the facet of the child nodes group. We find three groups of context words:

- {"algorithm(s)","model(s)","method(s)", "approach(es)", "technique(s)"...};
- {"application(s)","problem(s)","task(s)"...};
- {"measure(s)","metric(s)"...}.

Table 3: Performance of concept typing.

	Accuracy (True/False)
MajVot	0.874 (188/27)
MajVot+Grouping	0.963 (207/8)

Table 4: False type predictions in red.

Concept	Prediction	Ground Truth
topic_model	$*Method*	$*Method*
topic_models (synonym)	$*Problem*	$*Method*
mean_absolute_error	$*Metric*	$*Metric*
area_under_the_curve (sibling)	$*Method*	$*Metric*

(a) MajVot: 27 false predictions.

Concept	Prediction	Ground Truth
frequent_patterns	$*Object*	$*Object*
principal_components	$*Method*	$*Object*
information_gain	$*Metric*	$*Object*
cluster_analysis	$*Method*	$*Problem*

(b) MajVot+Grouping: 8 false predictions.

4 Experiments

We conduct experiments to answer three questions: (1) Are the three NIP modules effective in extracting hierarchical relations? (2) Does the hierarchy growth algorithm generate a hierarchy of better quality than existing methods? Are NIL nodes and redundant link removal necessary? (3) What does the result hierarchy look like?

4.1 Results on Three IE Components

M1: Scientific concept extraction. Table 2 shows examples of data science concepts the tools extracted. The learning module in *AutoPhrase* can segment words and phrases of good statistical features like high frequency. There is often no ambiguity when we lowercase them but the phrase lengths tend to be short. *SchBase* has a different philosophy: it looks for abbreviation expansions that could be long and of very low frequency. We show some case studies in Table 2. For result of

Table 5: Number of concepts of each type.

	$Problem	*$Method*	*$Object*	*$Metric*
Count (predicted)	52	104	9	50
Count (ground truth)	53	100	13	49

Table 6: Number of relations for each type.

	synonym	*sibling*	*ancestor*
# unique concept pairs	41	234	138
# extractions	1,966	1,379	381

AutoPhrase, some 1-gram and n-gram high quality phrase are in Table 2a. For results of *SchBase*, some acronyms and typical abbreviation expansions we selected are in Table 2b. With these two complementary tools, we harvest a collection of 215 data science concepts.

M2: Concept typing. Table 3 shows that the accuracy of concept typing (a 4-class classification task) is 0.874. Table 4a gives two of the 27 MajVot's false predictions. We observe that some synonym/sibling concept names like "topic_model" and "topic_models" have *inconsistent* predicted types due to the sparsity of their neighboring words. Therefore, we leverage the synonym/sibling relations discovered in the next subsection to group the related concept names together and determine their type based on the neighboring words of all the concepts in the group (called MajVot+Grouping). The accuracy is improved significantly to 0.963. Table 4b shows three of the 8 false cases among 215 predictions. Table 5 shows the number of concepts of each type we have for hierarchy induction.

M3: Hierarchical relation extraction. Table 6 gives the number of relation tuples we extracted for each type. The relation *synonym* has the highest number of extractions while *sibling* gives the most unique concept pairs.

4.2 Results on Hierarchy Quality Evaluation

Evaluation metrics. Based on the manually labelled parent-to-child relations, we evaluate the quality of the resulting hierarchy with three standard IR metrics, precision, recall, and F1 score, on extracting concept pairs that have a 0-hop path (i.e., synonyms), a 1-hop path (i.e., "parent-to-child" relation), and a 2-hop path (i.e., ancestor relation as parent's parent). A higher score means better performance.

Baseline method. It is not fair to compare with taxonomy construction methods because we are targeting a different problem, that is to generate a concept hierarchy of facets with three kinds of hierarchical relations. Therefore, we choose to compare with a hierarchy induction method, called *TAXI* (Panchenko et al., 2016), and we feed it with all the relations we mined so that we only compare on the performance of hierarchy induction algorithms. However, *TAXI* has no module to consider the sibling relations but we have the "NIL" mechanism. *TAXI* goes through all the relations several times, removes cycles, and links disconnected components to the root, while we consider the relation weights and generate the hierarchy in a growth manner for one scan. Therefore, compared with *TAXI*, *HiGrowth* is a more efficient algorithm on generating a facet concept hierarchy.

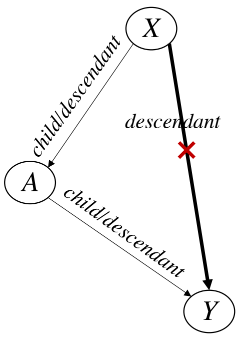
Table 7: Comparing *HiGrowth* with baselines on building hierarchy from data science literature.

Method	**Path**	**Precision**	**Recall**	**F1**
TAXI	0-hop	1.0	.5034	.6697
	1-hop	1.0	.4004	.5719
	2-hop	1.0	.1831	.3095
HiGrowth w/o "NIL"	0-hop	1.0	.5294	**.6923**
	1-hop	.9482	.4583	.6179
	2-hop	.9499	.3038	.4603
HiGrowth	0-hop	1.0	.5294	**.6923**
	1-hop	.9926	.5781	**.7307**
	2-hop	.9987	.5253	**.6885**

Quality analysis. As shown in Table 7, *HiGrowth* consistently outperforms *TAXI* on all three kinds of paths: it improves synonym detection by 3.4%, parent relation extraction by 27.8%, and 2-hop ancestor relation extraction by from 0.31 to 0.69. Actually, the *HiGrowth* variant that disabled the generation and removal of "NIL" node can still outperform *TAXI* because the hierarchy grows with relations from the most confident to the least confident. With the "NIL" nodes, *HiGrowth* improves the 1-hop relation by 18.3% and 2-hop relation by 49.6%. This shows that it is important to carefully consider the *sibling* relations.

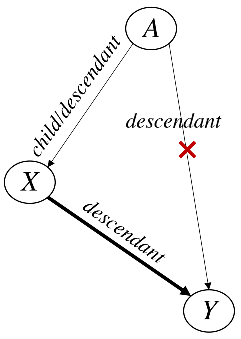
4.3 Results on Removing Redundant Links

Figure 7 presents redundant links that *HiGrowth* skipped or removed when adding a new relation *ancestor*(X, Y) for each of the three situations, respectively. The most common situation is that, we have *ancestor*(A, X) and *ancestor*(A, Y) in the hierarchy, and now we have a new link to specify the relation between X and Y, two descendants of A. If X is an ancestor of Y, we remove the redundant link *ancestor*(A, Y). We can see a few examples of the 93 redundant relations. A is a more gen-



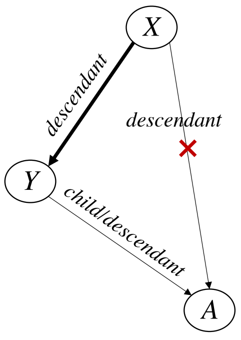
X	A	Y
data_mining	*classification*	*naïve_bayes*
data_mining	*classification*	*decision_trees*
data_mining	*classification*	*support_vector_ machines*
data_mining	*dimensionality_ reduction*	*principal_comp onent_analysis*
…	…	…
ensemble	*boosting*	*adaboost*

7



A	X	Y
data_science	*data_mining*	*clustering*
data_science	*machine_ learning*	*logistic_ regression*
data_science	*data_mining*	*outlier_ detection*
data_science	*machine_ learning*	*random_ forests*
…	…	…
machine_ learning	*supervised_ learning*	*neural_ networks*
classification	*decision_tree*	*CART*
classification	*decision_tree*	*C4.5*

93



X	Y	A
data_mining	*frequent_pattern _mining*	*fp-growth*
data_mining	*frequent_pattern _mining*	*apriori*

2

Figure 7: The redundant links that the *HiGrowth* algorithm removed during hierarchy construction.

eral (ancestor-level) concept. The frequency of A is often higher than the frequency of X or Y. The weights of *ancestor*(A, X) and *ancestor*(A, Y) are bigger than the weight of *ancestor*(X, Y). So the latter relation will be added to the hierarchy when the other two have been on the hierarchy.

4.4 Visualizing the Concept Hierarchy

Figure 8 presents the concept hierarchy that *HiGrowth* extracted from the Data Science publications. The hierarchy is not very large but still not visible in one page, so we enlarge three parts of the hierarchy, including (1) a set of concepts as the "measures" facet of "binary_classification," (2) the "applications" and "algorithms" facets of the concept "classification," and (3) the "algorithms" of "community_detection," the "techniques" of "matrix_factorization," and the "methods" of "feature_extraction" and "dimensionality reduction." We represent the relations of synonyms by adding different surface names for same entities in one node. For example, "topic_models" and "topic_model" are merged into one node in Figure 8 because they have the same semantic meaning.

5 Related Work

5.1 Scientific Concept Extraction

Scientific concept extraction is a fundamental task (Yu et al., 2019; Jiang et al., 2019). It has been widely studied on multiple kinds of text sources such as web documents (Parameswaran et al., 2010), business documents (Ménard and Ratté, 2016), clinical documents (Jonnalagadda et al., 2012), material science documents (Kim et al., 2017), and computer science publications (Upadhyay et al., 2018). The phrase mining technologies have been evolving from noun phrase analysis (Evans and Zhai, 1996) to recently popular representation learning methods (Mikolov et al., 2013; Pennington et al., 2014). Here we combined two methodologies that have been demonstrated to be effective in Science IE (Gábor et al., 2018).

5.2 Hierarchical Relation Extraction

There has been unsupervised methods on hypernym discovery and synonym detection (Weeds et al., 2014): In this work, we combine precise textual patterns, not only the syntactic patterns (Snow et al., 2005) but also the typed patterns (Nakashole et al., 2012; Li et al., 2018; Wang et al., 2019) to find synonyms and hypernyms. We consider hypernyms carefully as ancestor-to-descendant instead of parent-to-child relations. Synonyms are on the same node, and hypernyms are connected via one- or multi-hop path. Moreover, we extract the sibling relations which precisely describe the nodes on the same level. All the three types of relation tuples are important for inferring concept hierarchies.

5.3 Hierarchy Construction and Population

There are two kinds of hierarchy construction methods: one is taxonomy or ontology induction that infers "isA" relations by machine learning models (Kozareva and Hovy, 2010; Wu et al., 2012; Yang et al., 2015; Cimiano and Staab, 2005), and the other is topical hierarchy discovery that organizes phrases into topical groups and then infers hierarchical connections between the topical groups (Wang et al., 2015; Jiang et al., 2017). For the first kind of approaches, researchers used syntactic contextual evidence (Anh et al., 2014), belief propagation for population (Bansal et al., 2014), and embedding-based inference (Fu et al., 2014; Nguyen et al., 2014). For the second part, poincaré embedding and ontology embedding methods have been proposed to learn node representations from existing hierarchies (Nickel and Kiela, 2017; Chen et al., 2018).

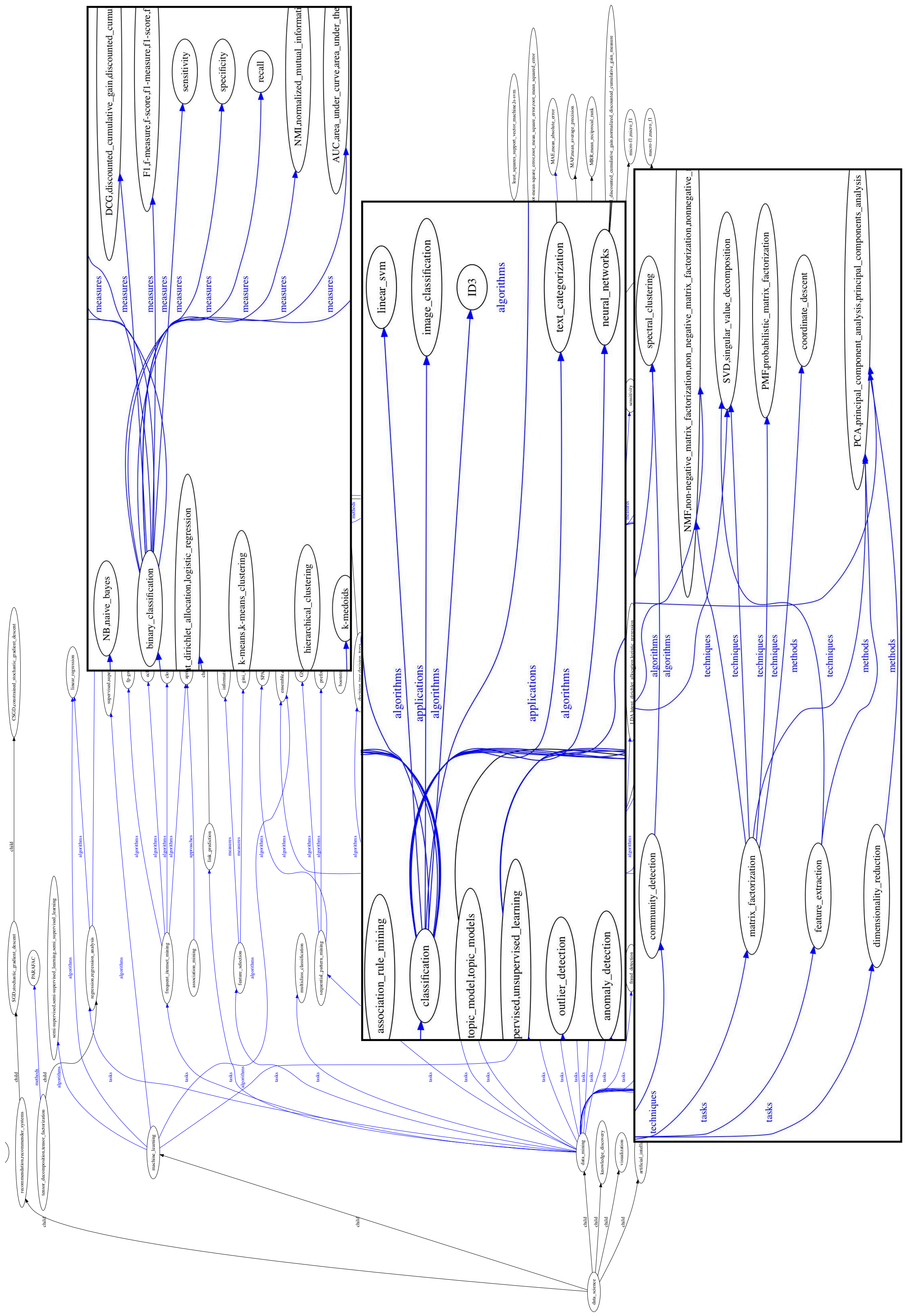
None of the existing approaches aimed at inferring parent-to-child relations based on the three types of hierarchical relations (i.e., synonym, ancestor-to-descendant, and sibling). We propose a novel hierarchy growth algorithm that addresses the issues of noisy, redundant, and invalid links.

6 Conclusions

This paper presented the *HiGrowth* method that constructs a faceted concept hierarchy from literature data. The major focus is on growing a hierarchy from three kinds of hierarchical relations that were extracted by pattern-based IE and weighted by their frequency. The hierarchy growth algorithm handles unnecessary, invalid and redundant links, even the relation set is noisy at the long tail.

Acknowledgements

This work was supported by Natural Science Foundation Grant CCF-1901059.



data_science
data_mining
knowledge_discovery
visualization
machine_learning
recommendation,recommender_systems
tensor_decomposition,tensor_factorization
child
tasks
algorithms
measures
applications
techniques
methods
association_rule_mining
classification
topic_model,topic_models
outlier_detection
anomaly_detection
community_detection
matrix_factorization
feature_extraction
dimensionality_reduction
linear_svm
image_classification
ID3
text_categorization
neural_networks
spectral_clustering
NMF,non-negative_matrix_factorization,non_negative_matrix_factorization,nonnegative_
SVD,singular_value_decomposition
PMF,probabilistic_matrix_factorization
coordinate_descent
PCA,principal_component_analysis,principal_components_analysis
NB,naive_bayes
binary_classification
k-means,k-means_clustering
hierarchical_clustering
k-medoids
DCG,discounted_cumulative_gain,discounted_cumu
F1,f-measure,f-score,f1-measure,f1-score,f
sensitivity
specificity
recall
NMI,normalized_mutual_informati
AUC,area_under_curve,area_under_the
SGD,stochastic_gradient_descent
CSGD,constrained_stochastic_gradient_descent
PARAFAC
regression,regression_analysis
linear_regression
frequent_itemset_mining
association_mining
feature_selection
multiclass_classification
sequential_pattern_mining
link_prediction

Figure 8: The resulting faceted concept hierarchy we extracted from Data Science publications, nodes mean the entities with different surface names (synonyms).

References

Eytan Adar and Srayan Datta. 2015. Building a scientific concept hierarchy database (schbase). In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 606–615.

Tuan Luu Anh, Jung-jae Kim, and See Kiong Ng. 2014. Taxonomy construction using syntactic contextual evidence. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 810–819.

Mohit Bansal, David Burkett, Gerard De Melo, and Dan Klein. 2014. Structured learning for taxonomy induction with belief propagation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1041–1051.

Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, and Carlo Zaniolo. 2018. On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 315–323. SIAM.

Philipp Cimiano and Steffen Staab. 2005. Learning concept hierarchies from text with a guided agglomerative clustering algorithm. In *Proceedings of the ICML 2005 Workshop on Learning and Extending Lexical Ontologies with Machine Learning Methods*.

David A Evans and Chengxiang Zhai. 1996. Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 17–24. Association for Computational Linguistics.

Ruiji Fu, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1199–1209.

Kata Gábor, Davide Buscaldi, Anne-Kathrin Schumann, Behrang QasemiZadeh, Haifa Zargayouna, and Thierry Charnois. 2018. Semeval-2018 task 7: Semantic relation extraction and classification in scientific papers. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 679–688.

Amit Gupta, Rémi Lebret, Hamza Harkous, and Karl Aberer. 2017. Taxonomy induction using hypernym subsequences. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1329–1338. ACM.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics-Volume 2*, pages 539–545. Association for Computational Linguistics.

Tianwen Jiang, Ming Liu, Bing Qin, and Ting Liu. 2017. Constructing semantic hierarchies via fusion learning architecture. In *China Conference on Information Retrieval*, pages 136–148.

Tianwen Jiang, Tong Zhao, Bing Qin, Ting Liu, Nitesh V Chawla, and Meng Jiang. 2019. The role of “condition”: A novel scientific knowledge graph representation and construction model. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1634–1642. ACM.

Siddhartha Jonnalagadda, Trevor Cohen, Stephen Wu, and Graciela Gonzalez. 2012. Enhancing clinical concept extraction with distributional semantics. *Journal of biomedical informatics*, 45(1):129–140.

Edward Kim, Kevin Huang, Adam Saunders, Andrew McCallum, Gerbrand Ceder, and Elsa Olivetti. 2017. Materials synthesis insights from scientific literature via text extraction and machine learning. *Chemistry of Materials*, 29(21):9436–9444.

Zornitsa Kozareva and Eduard Hovy. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 1110–1118. Association for Computational Linguistics.

Qi Li, Meng Jiang, Xikun Zhang, Meng Qu, Timothy P Hanratty, Jing Gao, and Jiawei Han. 2018. Truepie: Discovering reliable patterns in pattern-based information extraction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1675–1684. ACM.

Jiaqing Liang, Yi Zhang, Yanghua Xiao, Haixun Wang, Wei Wang, and Pinpin Zhu. 2017. On the transitivity of hypernym-hyponym relations in data-driven lexical taxonomies. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Ninghao Liu, Xiao Huang, Jundong Li, and Xia Hu. 2018. On interpretation of network embedding via taxonomy induction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1812–1820. ACM.

Pierre André Ménard and Sylvie Ratté. 2016. Concept extraction from business documents for software engineering projects. *Automated Software Engineering*, 23(4):649–686.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. Patty: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1135–1145. Association for Computational Linguistics.

Viet-An Nguyen, Jordan L Ying, Philip Resnik, and Jonathan Chang. 2014. Learning a concept hierarchy from multi-labeled documents. In *Advances in Neural Information Processing Systems*, pages 3671–3679.

Maximillian Nickel and Douwe Kiela. 2017. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347.

Alexander Panchenko, Stefano Faralli, Eugen Ruppert, Steffen Remus, Hubert Naets, Cédrick Fairon, Simone Paolo Ponzetto, and Chris Biemann. 2016. Taxi at semeval-2016 task 13: a taxonomy induction method based on lexico-syntactic patterns, substrings and focused crawling. In *SemEval-2016*, pages 1320–1327.

Aditya Parameswaran, Hector Garcia-Molina, and Anand Rajaraman. 2010. Towards the web of concepts: Extracting concepts from large datasets. *Proceedings of the VLDB Endowment*, 3(1-2):566–577.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Transactions on Knowledge and Data Engineering*, 30(10):1825–1837.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in neural information processing systems*, pages 1297–1304.

Prajna Upadhyay, Ashutosh Bindal, Manjeet Kumar, and Maya Ramanath. 2018. Construction and applications of teknowbase: a knowledge base of computer science concepts. In *Companion Proceedings of the The Web Conference 2018*, pages 1023–1030. International World Wide Web Conferences Steering Committee.

Chi Wang, Jialu Liu, Nihit Desai, Marina Danilevsky, and Jiawei Han. 2015. Constructing topical hierarchies in heterogeneous information networks. *Knowledge and Information Systems*, 44(3):529–558.

Xueying Wang, Haiqiao Zhang, Qi Li, Yiyu Shi, and Meng Jiang. 2019. A novel unsupervised approach for precise temporal slot filling from incomplete and noisy temporal contexts. In *The World Wide Web Conference*, pages 3328–3334. ACM.

Julie Weeds, Daoud Clarke, Jeremy Reffin, David Weir, and Bill Keller. 2014. Learning to distinguish hypernyms and co-hyponyms. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2249–2259. Dublin City University and Association for Computational Linguistics.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492. ACM.

Shuo Yang, Yansong Feng, Lei Zou, Aixia Jia, and Dongyan Zhao. 2015. Taxonomy induction and taxonomy-based recommendations for online courses. In *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 267–268. ACM.

Wenhao Yu, Zongze Li, Qingkai Zeng, and Meng Jiang. 2019. Tablepedia: Automating pdf table reading in an experimental evidence exploration and analytic system. In *The World Wide Web Conference*, pages 3615–3619. ACM.

Chao Zhang, Fangbo Tao, Xiusi Chen, Jiaming Shen, Meng Jiang, Brian Sadler, Michelle Vanni, and Jiawei Han. 2018. Taxogen: Constructing topical concept taxonomy by adaptive term embedding and clustering. In *KDD*.