HierCon: Hierarchical Organization of Technical Documents Based on Concepts

Keqian Li[†], Shiyang Li[†], Semih Yavuz[†], Hanwen Zha[†], Yu Su[¶], Xifeng Yan[†]
University of California, Santa Barbara[†]
Ohio State University, Columbus[¶]
{klee, shiyangli,syavuz,hanwen_zha,xyan}@cs.ucsb.edu su.809@osu.edu

Abstract—**In this work, we study the problem of automatically organizing the technical documents into different nodes in a given taxonomy. Unlike prior work on supervised hierarchical document categorization that relies on learning from labeled training data, which is expensive to obtain in closed technical domain and tends to stale as new knowledge and taxonomy evolves, we study this problem in a weak supervision setting, by leveraging semantic information from concepts. The core idea is to project both documents and taxonomy categories into a common *concept space*, where their fine-grained similarity can be easily and effectively computed. Experiments on real-world datasets from the field of computer science, physics & mathematics, and medicine show that the proposed method can consistently outperform a wide range of strong baselines by a significant margin.**

I. Introduction

The large volume of the scientific literature are becoming prohibitive: according to the 2018 International Association of Scientific, Technical and Medical Publisher's report [1], about 3 million journal articles are published every year with a 5% annual growth rate. Therefore, it is becoming more and more difficult to keep up with the scientific advancements for both researchers and practitioners. Advanced techniques for better understanding and organizing the scientific literature are in great demand. According to cognitive science studies [2], [3], a key management strategy for such information is to organize them into a hierarchical taxonomy, which has been widely used in the area of library study [4], internet directories[1][2], patents[3] and many others [5].

In this work, we study how to automatically organize scientific publications into a given hierarchical taxonomy. There are a number of previous methods for this problem focusing on general-domain documents, which mainly treat it as a supervised classification problem [6], [7], [8]. The main idea is to adapt standard classifiers to the hierarchical setting by incorporating similarity information of the nodes in the hierarchy into the classification model. But they require a significant amount of manually labeled document-class pairs, which is prohibitively expensive for scientific publications because the annotation can only be fulfilled by highly skilled domain experts. In addition, the science is very dynamic, so the taxonomy is also rapidly evolving [9], making it not economical to commit too much annotation effort to one static taxonomy. Another line of research [10] exploits distant supervision [11], but they assume a knowledge base which may not exist for many fields.

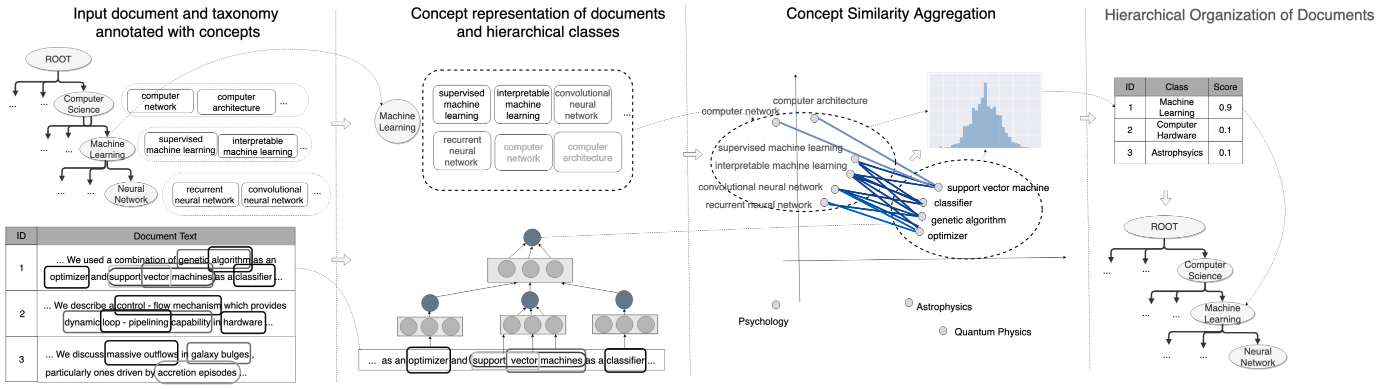
We propose to study the task of hierarchical document categorization with *weak supervision*, which only assumes very few labeled training examples. Because of the low annotation cost, it is also easy to adapt to different taxonomies. More specifically, given a hierarchical taxonomy and a set of documents, the goal is to categorize the documents into the categories in the taxonomy [4], where only a few labeled documents are provided for each category. The hierarchical nature of the problem makes it much more challenging than binary or multinomial classification: a classifier needs to respect the hierarchical structure among the categories.

There are a number of approaches to this problem. One may take a clustering approach, which detects coherent clusters from documents in an unsupervised fashion. Hierarchical information can be incorporated into the clustering process by using a hierarchical clustering algorithm [12], or by matching the clusters to the leaf nodes of the taxonomy. A more flexible approach is to treat each category as a keyword query, and leverage information retrieval techniques such as query expansion to retrieve the documents [13]. Finally, one could also repetitively apply simple flat unsupervised models [9], and concatenate their predictions together to obtain the final categorization in the hierarchical taxonomy.

We explore a vastly different categorization paradigm. It is based on the observation that technical documents are typically organized around *concepts*, which bear discriminative information about their topics (categories). For example, a database paper usually involves concepts such as "map and reduce" and "SQL", while a machine learning paper involves concepts like "statistical inference" and "convergence rate". Following this, we develop our concept based hierarchical document categorization framework called *HierCon*, which leverages concepts in technical documents to form semantic representations for both the documents and the taxonomy categories. We then derive the similarity between documents

[1]https://en.wikipedia.org/wiki/Yahoo!_Directory
[2]https://dmoztools.net/
[3]https://www.wipo.int/classifications/ipc/en/

[4] We may use the set of terms "organize", "categorize", "classify" and also use the set of terms "hierarchical label", "taxonomy node", "hierarchical category" interchangeably depending on the context.


Input document and taxonomy annotated with concepts
ROOT
Computer Science
computer network
computer architecture
Machine Learning
supervised machine learning
interpretable machine learning
Neural Network
recurrent neural network
convolutional neural network
Concept representation of documents and hierarchical classes
Machine Learning
supervised machine learning
interpretable machine learning
convolutional neural network
recurrent neural network
computer network
computer architecture
... as an optimizer and support vector machines as a classifier ...
Concept Similarity Aggregation
computer architecture
computer network
supervised machine learning
interpretable machine learning
convolutional neural network
recurrent neural network
support vector machine
classifier
genetic algorithm
optimizer
Psychology
Astrophysics
Quantum Physics
Hierarchical Organization of Documents
ROOT
Computer Science
Machine Learning
Neural Network

ID	Document Text
1	... We used a combination of genetic algorithm as an optimizer and support vector machines as a classifier ...
2	... We describe a control - flow mechanism which provides dynamic loop - pipelining capability in hardware ...
3	... We discuss massive outflows in galaxy bulges, particularly ones driven by accretion episodes ...

ID	Class	Score
1	Machine Learning	0.9
2	Computer Hardware	0.1
3	Astrophysics	0.1

Fig. 1: HierCon Framework Overview.

and categories using their concept level representation, based on which we perform hierarchical categorization. Unlike previous approaches that employ model specific engineering to incorporate hierarchy information into the category, for example, by imposing regularization constraints over classifiers at different levels of hierarchy [6] or by encoding the similarity into model parameters [7], we propose to represent the categories as *distribution* over concepts, which allows for flexible combination to express the similarity. Furthermore, by projecting both the categories and documents into the space of concept level representation, their relevance becomes more explicit and easier to capture.

To derive the concept level representation, we employ state-of-the-art concept mining techniques [14] to comprehensively discover concepts and their occurrence locations in the documents, and learn their vectorized semantic embedding representation without supervision signal. For example, we are able to mine concept occurrences such as ”vector machine” and ”support vector machine”, which may overlap with each other. The task is then to select true concepts from candidate concepts to represent the document. To address this, we propose a novel, adaptive concept level document representation based on the hierarchical neural attention mechanism [15], which models the two-level-decision as a natural hierarchical process, incorporating important signals about phrase quality [14], [16] and importance [17] into the modeling process, and is able to dynamically adjust the concept representation based on downstream performance.

However, there is still a large gap between the concept semantics and the task of discriminatively associating documents with the hierarchical categories. If there is a large amount of labeled training data, we could simply treat this as a standard supervised classification problem. Without that luxury, however, this task becomes more challenging. We propose a novel approach to compute concept based relevance by exploiting their inner structure. The main idea is that many concepts in the document bear discriminative information and can be easily associated with taxonomy categories. Motivated by this, we propose a principled approach for aggregating the discriminative information between all concepts in the document and category’s concept representation at all different similarity strength, in order to obtain document-category relevance and perform classification.

We evaluate our method on three real-world datasets from the fields of computer science, physics & mathematics and medicine, and compare it with a wide range of strong baseline methods of different nature. The results show that our method consistently outperforms all the baseline methods by a significant margin.

In summary, our contribution is three-fold:

- We study a novel problem of hierarchical categorization of technical documents according to a target taxonomy, without a large amount of labeled training data or existing knowledge bases.
- We propose a novel concept based approach that represents both the taxonomy categories and the documents using concepts mined from the entire corpus, which can be effectively used to compare their similarities and categorize documents accordingly.
- We comprehensively evaluate our approaches in comparison with the state-of-the-art hierarchical classification methods over three real-world datasets in the fields of computer science, physics & mathematics and medicine.

II. Related work

A. Hierachical text classification

Previous work on hierarchical document categorization [8] usually takes a supervised classification approach, where hierarchy structure is taken into account by enforcing similarities between adjacent classifiers [18], [19], [6]. Following this, the dataless hierarchical classification approach [10] was proposed where they employ a large knowledge base such as Wikipedia to generate semantic representations of documents and labels and further apply them into the hierarchical setting using a top-down, or bottom up approach. The above approaches either heavily rely on the availability and quality of labeled training set for each specific class, or expect the corpus is covered by an external knowledge base, both of which are different from our setting. A very recent work [20] has similar settings to ours, where given a few labeled training data, they first generate a representation of each class, use it to generate more pseudo-documents in order to train a classifier, and then bootstrap on unlabeled data. Their method requires significantly more parameters to tune and also relies heavily on the quality of training data.

B. *Entity aware representation*

Another line of related work is on concept/entity aware representation, which utilizes information from a knowledge base, such as entity description, entity type or links to other entities to perform query expansion [21], [22], and improves relevance computation [23], or ranking [24]. Explicitly annotated entities in documents have been utilized to represent text [25], [26], which are then used to derive features to train a ranking model. However, directly adapting these information retrieval approaches to hierarchical classification setting, where one needs to assign each document into the correct category, is rather challenging as the query needs to cover the content of the entire scientific field.

C. *Classification without explicitly labeled training data*

Yet another set of related work is concerned with the task of performing classification without explicit labeled training data. The common characteristics of these approaches is to exploit the semantic meaning of the label. For example, in the computer vision research community, this is known as zero-shot learning, which learns the semantic embedding of labels [27], and inputs [28], [29], [30], or both [31], [32]. These approaches mostly follow a representation based approach, which relates unseen classes and seen ones by extracting features of each class to transform them into vector representations, and directly learns a classifier between data and feature vectors. On the other hand, unsupervised neural embedding methods could be potentially leveraged to solve the unsupervised categorization task, where embedding of single word [33], [34], sentence [35], and relation [36] can be learned to reveal the semantics to support downstream analytical tasks. [9] is among the first to study the novel problem of unsupervised categorization, which builds upon neural embedding approach, and learns the category attribution of the documents based on a compressed version of the word/concept embedding.

III. THE HIERCON FRAMEWORK

We formulate the weakly supervised hierarchical categorization task as follows. The first set of input is a corpus of plain text documents $\mathcal{D}$, each document $d \in \mathcal{D}$ being a sequence of words in the form of $w_1 w_2 \dots w_{|d|}$. Secondly, we're given a set of target categories $\mathcal{Y}$, organized into a tree structure, so that each category $y \in \mathcal{Y}$ is associated with one parent $P(y) \in \mathcal{Y}$, and a set of ancestors $\mathcal{A}(y)$, including its parent, grandparent, and so on. The task is then to associate each document $d \in \mathcal{D}$ with a relevant label $y \in \mathcal{Y}$. We study this in a weakly supervised setting, where users are allowed to provide a set of labeled training data, as a set of l (d_i, y_i) pairs, $l \ll \mathcal{D}$.

Our proposed approach is illustrated in Figure 1. In order to leverage *concepts* to represent documents and categories,

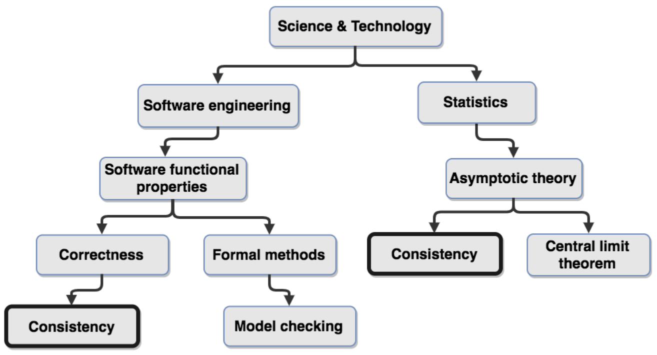


Fig. 2: An example taxonomy showcasing the importance of *path semantics*. The taxonomy node "Consistency" under Software engineering and another node "Consistency" concept under the Statistics can be distinguished by their ancestors, descendents and neighbors.

we first perform concept mining [14], i.e. identifying meaning bearing units such as "support vector machine" and "generative adversarial network" from text. Specifically, it will generate a concept vocabulary $\mathcal{V}$ as a set of concepts that occur in the corpus, and further recognize the (possibly overlapping) occurrences of each concept $c \in \mathcal{V}$. Using techniques described in section 4, we will generate the concept level representation $\vec{\varphi}(\cdot) : \mathcal{D} \to \mathbb{R}^{|\mathcal{V}|}$, that maps each document d in $\mathcal{D}$ to a vector, where each dimension denotes the associated concepts. The results will be a set of concepts that occurs in it along with their associated attention weights, as a $\mathcal{V}$ dimensional vector.

Associating target categories with relevant concepts in the concept vocabulary $\mathcal{V}$ based on string similarity, we can derive a basic concept representation for each label [9]. We encode it as a $|\mathcal{V}|$ dimensional weight vector $\phi^{(b)}(y)$ over all concepts $\mathcal{V}$. As an example, a node with the name "Consistency" will be associated with concepts such as "consistency" and "consistency principle", and such relevant concepts will have high value in the corresponding dimensions in its representation vector. However, solely relying on the basic semantics fails to capture hierarchy information and could lead to potential ambiguity, as illustrated by the example below.

Example 1 (concept representation of category nodes) Consider the hierarchical taxonomy which we want to categorize documents into shown in Figure 2. The taxonomy is simplified but manifests the following key observations: (1) Many interesting and concrete concepts are buried deep in the hierarchy, which is not directly contained in the label names of their parent nodes, but are strongly indicative from their parent concepts. (2) Conversely, the location of a node in the taxonomy, e.g. where its parent and grandparent are, is essential for determining the actual content it represents. For example, the category "Consistency" under the "Software engineering - Software functional properties - Correctness" hierarchy, and the category "Consistency" under the "Statistics, Asymptotic theory" hierarchy have significantly different meanings.

Motivated by the first observation in the example, we introduce *aggregated representation* $\phi^{(a)}(y)$, which enriches the semantics of each node with the basic semantics of all the taxonomy nodes in its sub-tree. It is defined recursively as follows

$$\phi^{(a)}(y) \triangleq \begin{cases} avg(\{\phi^{(a)}(y')|y' \in C(y)\} \cup \phi^{(b)}(y)) & C(y) \neq \emptyset \\ \phi^{(b)}(y) & \text{o.w.} \end{cases} \tag{1}$$

where $C(y) = P^{(-1)}(y)$ denotes the set of children of node y, and avg denotes a specific function for averaging the value. In this work we simply use the arithmetic mean.

Motivated by the second observation, we introduce *path semantics* $\phi^{(p)}(v)$ for each node $y \in \mathcal{Y}$, as an average over all the aggregated semantics of the current node y and all its ancestors $\mathcal{A}(y)$[5], which we will use as the final node representation. Specifically,

$$\phi(y) \triangleq \phi^{(p)}(y) \triangleq avg(\{\phi^{(a)}(y')|y' \in \mathcal{A}(y)\}, \phi^{(a)}(y)) \quad \forall y \in \mathcal{Y} \tag{2}$$

As a result, concepts that are closer in the tree share similar "tails" in their concept representation.

Given the concept representation of taxonomy nodes $\phi(y)$ and documents $\varphi(d)$, the task is to learn to measure the similarity $\mathcal{S}(\phi(y), \varphi(d))$ based on their concept representations. One possible choice is to follow traditional top-down hierarchical classification, to sequentially make decisions on which child from the current node one should descend to, possibly incorporating future delayed awards using reinforcement learning [37]. However this will make the model more sensitive to the amount of labeled training data and therefore undesirable for the weakly supervised settings. In this work instead, we follow a big-bang approach [8] that classifies each document against all nodes at the same time, and predict the label as the one maximizing the similarity derived from path semantics. Specifically, for model inference, we obtain the prediction $\hat{y}(d)$ for each document d by

$$\hat{y}(d) \triangleq \arg\max_{y \in \mathcal{Y}}(\mathcal{S}(\phi(y), \varphi(d))) \qquad \forall d \in \mathcal{D} \tag{3}$$

while we use the cross-entropy loss [6] as the objective function to minimize for model training where the relevance scores $\mathcal{S}(\phi(y), \varphi(d))$ are used as the corresponding logits over classes $y \in \mathcal{Y}$.

The overall procedure is summarized in Algorithm 1. Given a large text corpus $\mathcal{D}$, it first learns the concept vocabulary $\mathcal{V}$ as well as the concept level representation $\varphi(d)$ of each document $d \in \mathcal{D}$. Then, using the input hierarchical tree $\mathcal{T}$, it derives the semantic representation of each label by first computing basic semantics with respect to each label, followed by a bottom up pass that obtains the aggregated semantics using its child nodes recursively. These aggregated semantics are then passed down in a top-down pass, to obtain the path semantics $\varphi_v^{(p)}$ for each label. Based on the semantic representations of labels and documents, a relevance function can then be computed. Finally, for each document, we perform categorization by choosing the top K category labels for each document d that have the highest relevant scores.

[5]We tweak the definition of avg a little. So when called with two arguments $avg(\cdot, \cdot)$, it will be computed by taking the average over each argument, and averaging the result together: in other word, the second argument, in our case the basic representation of the current node, will take exactly half of the weight.

[6]https://en.wikipedia.org/wiki/Cross_entropy

Algorithm 1 Categorization Framework

```
Input: a corpus of plain text documents $\mathcal{D}$, set of target categories organized as a rooted tree $\mathcal{T}$, number of labels to predict for each document
Output: : top $K$ labels $\mathcal{L}(d)$ for each document $d \in \mathcal{D}$
Obtain concept vocabulary $\mathcal{V}$, concept representation $\{\varphi(\vec{d})|d \in \mathcal{D}\}$, and concept embedding $\{\theta_v|v \in \mathcal{V}\}$ from raw corpus $\mathcal{D}$
for $v \in \mathcal{T}$ do
    Obtain basic semantics $\varphi_v^{(b)} \in \mathbb{R}^{|\mathcal{V}|}$ based on the concept vocabulary $\mathcal{V}$ and the label name $v$
end for
for $v \in \mathcal{T}$ do
    compute basic semantics $\varphi_v^{(a)}$ according to (1)
end for
for $v \in \mathcal{T}$ do
    compute path semantics $\varphi_v^{(p)} \in \mathbb{R}^{|\mathcal{V}|}$ according to (2)
end for
compute relevance scoring function $\mathcal{S}(\phi(y), \varphi_d^{(p)}||\{\theta_c, c \in \mathcal{C}\})$
return top $K$ labels $c \in \mathcal{C}$ for each document $d \in \mathcal{D}$ that has the highest relevance scores according to $\mathcal{S}(\phi(y), \varphi_d^{(p)}||\{\theta_c, c \in \mathcal{V}\})$
```

The advantages are 3-folded: First, it can be directly computed compared to reinforcement learning based sequential approach, while flexible enough (by possibly changing the avg computation) to model the influence of class hierarchy into the node representation. It also allows the decision of assigning to leaf node vs. internal node: a document is assigned to leaf if it is more similar to a specific sub-field; Or if it is equally similar to them, and it stays at a more general level and is assigned to common parent node. Furthermore, it is interpretable and naturally provides not only the most likely category, but also the secondary category, third category, besides its primary one as auxiliary information.

IV. Concept representation for documents

In this section we discuss our approach for obtaining the concept representation $\vec{\varphi}(d)$ of each document $d \in \mathcal{D}$, where we follow a two-step approach: First, we pre-process the data and follow previous approaches in concept mining [14] to generate a large pool of potential concepts as candidates, which may contain false positive ones, or ones that are less important to the major topic of a document. For example, in the text shown in Figure 3, whereas both "vector machine" and "support vector machine" are recognized as concept candidate, we may want to only select "support vector machine" as

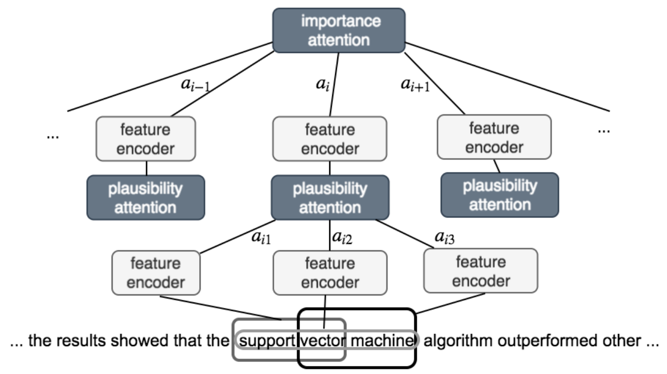


Fig. 3: Illustration of the hierarchical attention mechanism.

the correct occurrence, and furthermore select this as one of the central concepts in the document to be used in its representation.

To formalize this, we follow the assumption from entity recognition literature [38] and require that the textual occurrences of these concepts to not overlap with each other. The problem can then be described as: Given a document d as a sequence of words, the result of concept recognition will be to group these words into a sequence of super-concepts, i.e. overlapping regions, $u_1, u_2, u_3, \dots, u_l$, where each u_i contains a set of candidate concepts $c_{i1}, c_{i2}, \dots, c_{i|u_i|}$ that overlap with each other, and l is the number of super-concepts in the document. Our goal, then, is first to disentangle from the overlapping candidates and select valid concepts, and then determine important concepts from them and output the final representation of a document $\varphi(d) \in \mathbb{R}^{|\mathcal{V}|}$, as a categorical distribution over the vocabulary of concepts.

We propose an adaptive concept representation architecture based on the hierarchical attention mechanism [15], which directly combines noisy concept candidates into the final document representation in an end-to-end fashion, that dynamically adjusts the representation based on downstream performance. The hierarchical attention mechanism is shown in Figure 3. It works as follows: inside each super-concept, the model will choose to attend to one of its most probable candidates; And based on the choice of concept candidates from each super-concept, a second-level attention is drawn to select the most important super-concepts.

At the validity level, the following information features are captured and fed as input to the neural network

- **Grammatical significance** We count the number of the candidates occurred standalone as detected by grammatical pattern extractor, pre-trained chunking model and entity extraction model.
- **Statistical significance** We use the score outputted by the statistical significance based phrase mining tool Autophrase [16].
- **Context significance** We use the embedding based quality measure feature from ECON [14].

Formally, the attention score a_{it} for each $c_{it}, 1 \le t \le |u_i|$, inside each super-concept $u_i, 1 \le i \le l$, can be obtained by transforming the feasibility features for each candidates f_{ij} through transformation and softmax function

$$a_{it}^{pre} = f_{it} \cdot W_f \tag{4}$$

$$a_{it}^{s} = \frac{exp(a_{it}^{pre})}{\sum_t (exp(a_{it}^{pre}))} \tag{5}$$

Here W_f is the model parameters for weighting different feasibility features.

By the selection of feasibility features, our network will select important super-concepts and the following information are incorporated:

- **Occurrence semantics**: We adopt the occurrence location based feature, specifically, the first occurrence location of each super-concept divided by the number of super-concepts [17].
- **Span semantics**: We count the total number of words that super-concept spans. The intuition is that super-concept with more words are likely the one authors want to elaborate on and therefore may be more important.
- **Section semantics**: We check whether a super-concept occurs in specific logical sections in the document [17]. Specifically, in our case, we will assign the feature 1 if it occurs in the title and 0 otherwise.

The formalization of super-concept level attention is similar to the overlpapping ones, Given the importance features f_i for each super-concept i, $1 \le i \le l$, we have

$$a_i^{pre} = f_i \cdot W_i \tag{6}$$

$$a_i = Z(\{a_i^{pre} | 1 \le i \le l\})_i \tag{7}$$

$$\tag{8}$$

where W_i is the weight parameters associated with importance features, and the gating operation $Z(\cdot)$ refers to the gated-softmax operation described above.

The concept representation $\varphi(d)$ for each document $d \in \mathcal{D}$, as a $|\mathcal{V}|$ dimensional vector, will gather the attention weights distributed to each concept. Specifically, the value of $\varphi(d)$ for each dimension corresponding to a specific concept c, $(\varphi(d))_c$ will be computed as

$$(\varphi(d))_c = \sum_{i,t} a_i \cdot a_{it} \cdot \mathbb{I}(c_{ij} = c) \qquad \forall c \in \mathcal{V} \tag{9}$$

V. Concept Based Relevance

In this section we describe the model for computing the relevance $\mathcal{S}(\varphi(d), \phi(y))$ between each pair of taxonomy node and document $(y, d) \in \mathcal{Y} \times \mathcal{D}$, based on their concept representations.

The approach of measuring the similarities between labels and documents is relevant to several lines of previous work. Zero-shot learning in computer vision embeds labels and input instances into a latent embedding space, and tries to learn a compatibility function between the embedding vectors [39]; Information retrieval extracts features from documents and queries, and build a model to predict their relevance [40], [41]. These models are still supervised in nature that are trained on seen classes to predict the unseen ones. There are also models that leverage an external knowledge base to derive vector representation and relevance. None of the above are applicable to our scenario, where both training examples and knowledge base coverage are scarce.

To address this, we propose a novel *similarity aggregation* framework that exploits the interaction between the individual concepts in a document and label's concept representation to learn a direct relevance signal and aggregate them together in order to obtain document-label relevance and perform classification. Formally, let $S(c_i, c_j), 1 \leq i, j \leq |\mathcal{V}|$ denote the pre-computed concept-wise similarity function that is used to measure the similarity based on the learned embeddings, which we implement as cosine similarity in this work. Our goal is then to aggregate these similarities together using the document and taxonomy node's weights, in the form of

$$\mathcal{S}(\varphi(d), \phi(y)) = agg(\{S(c_i, c_j), 1 \leq i, j \leq |\mathcal{V}|\}, \varphi(d), \phi(y)) \tag{10}$$

where agg is the aggregation function that we will elaborate later.

Formally, assume that all the concepts are arranged into a list, $c_1, c_2, \ldots, c_{|\mathcal{V}|}$, and that we have trained the embedding vectors for each concept based on the concept mining [14].

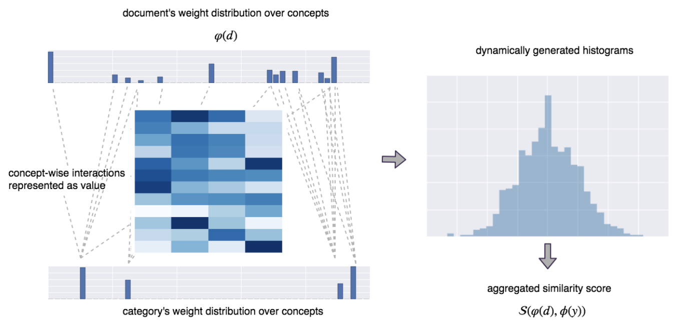
In order to obtain direct relevance signal from concept-wise interaction, we leverage the embedding lookup operation, which *learns* the relevance signal based on the raw concept similarities. Inspired by the idea of bin-pooling [40], which uses raw similarity scores between words in query and document as features for downstream tasks, our approach will also learn the relevance signal as a function of the embedding similarity. The architecture is shown in Figure 4. It works as follows, we divide the range of similarity into K discrete bins, with the kth bin covering the range $[st_k, end_k)$, $1 \leq k \leq K$, so that the similarity of each pair of concept $(c_i, c_j) \in \mathcal{V} \times \mathcal{V}$ will be mapped to a specific bin $k_{i,j}$. Then we *embed* each bin into a (arbitrarily) learned representation using the embedding parameter $\omega \in \mathbb{R}^K$, whose k-th dimension describes the relevance signal for the k-th bin. Using the embedding parameters, each concept pair (c_i, c_j) will be mapped to a representation, which directly corresponds to its relevance signal, $M(i, j|\omega)$, as

$$M(i, j|\omega) \triangleq \omega_{k_{i,j}} \qquad \forall 1 \leq i, j \leq |\mathcal{V}| \tag{11}$$

The task then becomes aggregating the mapped relevance signals together. In this work, we weigh the concept pairs $\{M(i, j|\omega)|1 \leq i, j \leq |\mathcal{V}|\}$ directly with the document's weight towards the ith concept, $(\varphi(d))_i$, and the taxonomy nodes' weight towards the j th concept, $(\phi(y))_j$, and obtain the relevance as

$$\mathcal{S}(\varphi(d), \phi(y)|\omega) \triangleq \sum_{i,j} (\varphi(d))_i \cdot (\phi(y))_j \cdot M(i, j|\omega) \tag{12}$$

The above model works as a dynamic version of histogram computing, where instead of using the similarity historgram as a fixed static feature for downstream tasks, the histogram is dynamically generated based on learnable weight distribution. We denote this as "dynamic bin pooling", which is able to adaptively attend to specific concepts, while allowing arbitrary interpretation of relevance signal from the raw different similarity strength.



document's weight distribution over concepts
$\varphi(d)$
dynamically generated histograms
concept-wise interactions represented as value
aggregated similarity score
$\mathcal{S}(\varphi(d), \phi(y))$
category's weight distribution over concepts
$\phi(y)$

Fig. 4: Dynamic bin-pooling for similarity aggregation.

Although in the above process, the binning operation is non-differentiable in general and prevents us from directly readjusting the concept similarity function $< c_i, c_j >$, it is, however, clear that the aggregation is differentiable in the concept weights and therefore allows us to fine-tune both the attention weights in document's concept representation W_f, W_i and parameters for the similarity interpretation ω in an end-to-end manner.

Gradient path saving However, directly optimizing the similarity aggregation model by current hardware may be hard due to the large number of concept interactions and the model parameters that are influenced by them during gradient computation. For example, a naive backpropagation would generate $\Omega(K|\mathcal{Y}||\mathcal{V}|^2)$ gradients which can easily exceed the memory capacity of modern GPU. Even if we store the concepts sparsely, the set of all concepts in the category's concept representation could be still quite large, and we need to consider the relevance for all y in $\mathcal{Y}$ during learning/prediction. To address this, we propose a *Gradient path saving* approach, which explores the best path for propagating the gradients based on the re-parameterization trick. Specifically, it can be shown that

$$\frac{\partial \mathcal{S}(\varphi(d), \phi(y)|\omega)}{\partial \omega_k} = \sum_{i,j} (\varphi(d) \otimes \phi(y)) \odot M_k \tag{13}$$

where $\otimes$ denotes vector outer-product, and each entry in $M_k(i, j)$ stores up the look-up results of $M_(i, j)$ for the kth bin, as

$$(M_k)_{i,j} = \mathcal{I}(st_k \leq < c_i, c_j > < end_k) \tag{14}$$

By pre-computing $(\varphi(d) \otimes \phi(y)) \odot M_k$ and treating it as fixed value, the amount of gradient computation can be cut down to $O(|\mathcal{Y}|)$. Similarly, the gradient for the document's concept weight $\varphi(d)_i, \forall i \in |\mathcal{V}|$ (or labels' weight) can also be reduced to $O(|\mathcal{Y}|)$ allowing for efficient implementation even on a single GPU card.

Monotonicity enforcement A common mode of failure for learning the bin weight $\{\omega_k, 1 \leq k \leq K\}$ is to have

TABLE I: Dataset Statistics

	Computer Science	Physics & Math	Medicine
# docs	47K	127K	55K
# words	9M	22M	9M
taxonomy size	31	28	17
taxonomy height	3	5	3

them ”scatter around”, where lower similarity strength may have even higher bin-weight than those with higher similarity strength. To ensure that the similarity weights have meaningful values, we employ a monotonicity enforcement, which enforces the weights for different similarity strengths to be monotonically non-decreasing via a activation function. Again, through re-parameterization, we store the *difference* of strength, as ω'_k, and each time, the kth bin-weight ω_k will be obtained as a computed value, using the ReLU activation function to enforce their monotonicity

$$\omega_k = \sum_{1 \le j \le k} \text{ReLU}\ (\omega_j^{'}) \qquad \forall k, 1 \le k \le K \tag{15}$$

Therefore, only the positive weight differences will be counted, and the network will learn to increase the bin-weight for higher similarity strength, or keep it the same.

Since our similarity aggregation model only learns parameters ω_k for different concept-wise similarity strength and the feature weight W_f, W_i for the concept-level features, we can efficiently tune these parameters using quite few labeled training documents, because each such document-label pair corresponds to a much large number of concepts and concept similarities pairs that we train our parameters on. In addition, because it makes prediction based on the concept level information, it is not limited to knowledge about specific classes and is naturally transferable to unseen classes labels.

VI. Experiment

In this section, we evaluate the proposed methods with extensive experiments across several technical domains and demonstrate its efficiency and effectiveness.

A. Experiment settings

1) Computing environment: All the model training and evaluation pipeline are conducted on a lab server with with 3 GeForce RTX 2080 GPU card and 2 6-core Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz CPU with 12GB memory. The longest run took less than 1 hour.

2) Datasets: We have collected the following corpus, covering the domain of computer science, physics & mathematics and medicine. The statistics of these datasets are summarized in Table I, and the details are described below.

Computer Science The Computer Science corpus is obtained by crawling paper abstracts from arXiv.org under the top level category ”computer science”, and aligning the arxiv’s category with aminer.org’s academic conference classification[7].

[7]https://aminer.org/ranks/conf

Physics & Mathematics The Physics & Mathematics corpus is also obtained from arXiv.org, by crawling paper abstracts under the top level category ”physics” and ”mathematics”, and aligning them with the math subject headings[8] and the physics subject headings[9],

Medicine The Medicine corpus is obtained by crawling abstracts from Pubmed[10], and we directly take the top 3 level of the Medicine subject headings[11] under the top level category ”Organisms”, ”Analytical, Diagnostic and Therapeutic Techniques, and Equipment” and ”Psychiatry and Psychology” as the taxonomy.

3) Compared methods: We compare our method with a wide range of state-of-the-art ones, as described below:

WeSHClass [20] is a state-of-the-art approach for weakly supervised hierarchical classification. It first generates a set of pseudo-documents for each class based on supervised signal such as labeled documents to train the classification model, then bootstrap on unlabeled data. We follow the author’s implementation[12] and use their recommended settings.

Dataless refers to the hierarchical dataless classifcation approach, which is [10] is a state-of-the-art distant supervision based method in the hierarchical classification settings. It works by first obtaining vector representations of documents and class labels by its similarity with Wikipedia articles [42], and associate documents based on the class labels based on its vector representation. We follow the original dataless implementation classification from the authors[13], and apply it with a bottom-up scheme.

Pretrain BERT is the currently the state-of-the-art deep learning approach for downstream NLP tasks, which leverage very deep self-attention based architecture with weights pretrained on large training corpus. Specifically, we add an classification layer added upon the first input token ([CLS]) from the last transformer layer, same as the experiment used for GLUE classification tasks [43], fine-tune all the layers and report the test accuracy under the best hyper-parameter settings.

Hierarchical SVM [19] augments SVM classifier with hierarchical information with a top-down paradigm [8], where training documents for child nodes are included in their ancestors.

UNEC [9] is a state-of-the-art unsupervised text categorization method that extracts concepts by segmenting the corpus into phrases and then learns a concept embedding graph, where similarity to classes are propagated. We adopt an alternative, personalized page rank approach[14] to propagate the similarity on the concept graph.

Hiercon refers to our proposed approach. We utilize the concept mining technique ECON [14] to obtain concepts for

[8]https://www.maa.org/press/periodicals/loci/joma/subject-taxonomy
[9]https://physh.aps.org/
[10]https://www.ncbi.nlm.nih.gov/pubmed/
[11]https://www.nlm.nih.gov/mesh/meshhome.html
[12]https://github.com/yumeng5/WeSHClass/
[13]https://cogcomp.org/page/download_view/Descartes
[14]https://networkx.github.io/documentation/networkx-1.10/reference/generated/networkx.algorithms.link_analysis.pagerank_alg.pagerank.html

TABLE II: Overall performance comparison measured by prediction accuracy. The top-K prediction score among all possible taxonomy nodes are recorded and compared against the ground truth according to both flat classification accuracy and taxonomy tree based accuracy

		WeSHClass		Pretrain BERT		Hier-SVM		Dataless		UNEC		Hiercon	
Dataset	Accuracy Measure	*Flat*	*Tree*	*Flat*	*Tree*	*Flat*	*Tree*	*Flat*	*Tree*	*Flat*	*Tree*	*Flat*	*Tree*
Computer Science	Top 1	0.3082	0.4453	0.4354	0.6207	0.1095	0.4156	0.2094	0.5453	0.1742	0.3502	0.7927	0.8763
	Top 3	0.4984	0.6672	0.6665	0.8676	0.2894	0.4747	-	-	0.3405	0.6633	0.9486	0.9613
	top 5	0.6004	0.7870	0.7728	0.9305	0.3685	0.6758	-	-	0.4358	0.8072	0.9803	0.9846
Physics & Mathematics	Top 1	0.3551	0.7100	0.3292	0.7262	0.0508	0.0555	0.2403	0.6229	0.2507	0.4149	0.7223	0.9270
	Top 3	0.6086	0.8692	0.6355	0.8450	0.0605	0.0605	-	-	0.4939	0.7329	0.9450	0.9791
	top 5	0.7292	0.9233	0.7985	0.9061	0.0608	0.0710	-	-	0.6157	0.8444	0.9787	0.9871
Medicine	Top 1	0.2478	0.7208	0.3550	0.7943	0.0000	0.5491	0.1641	0.6193	0.3275	0.5130	0.5296	0.8375
	Top 3	0.5562	0.8563	0.6775	0.9394	0.0512	0.7500	-	-	0.5932	0.8347	0.8399	0.9146
	top 5	0.7249	0.9567	0.8145	0.9805	0.3661	0.7903	-	-	0.7361	0.9256	0.9437	0.9864
Average (Top 1)	-	0.3037	0.6254	0.3732	0.7137	0.0534	0.3401	0.2046	0.5958	0.2865	0.5331	0.6815	0.8803

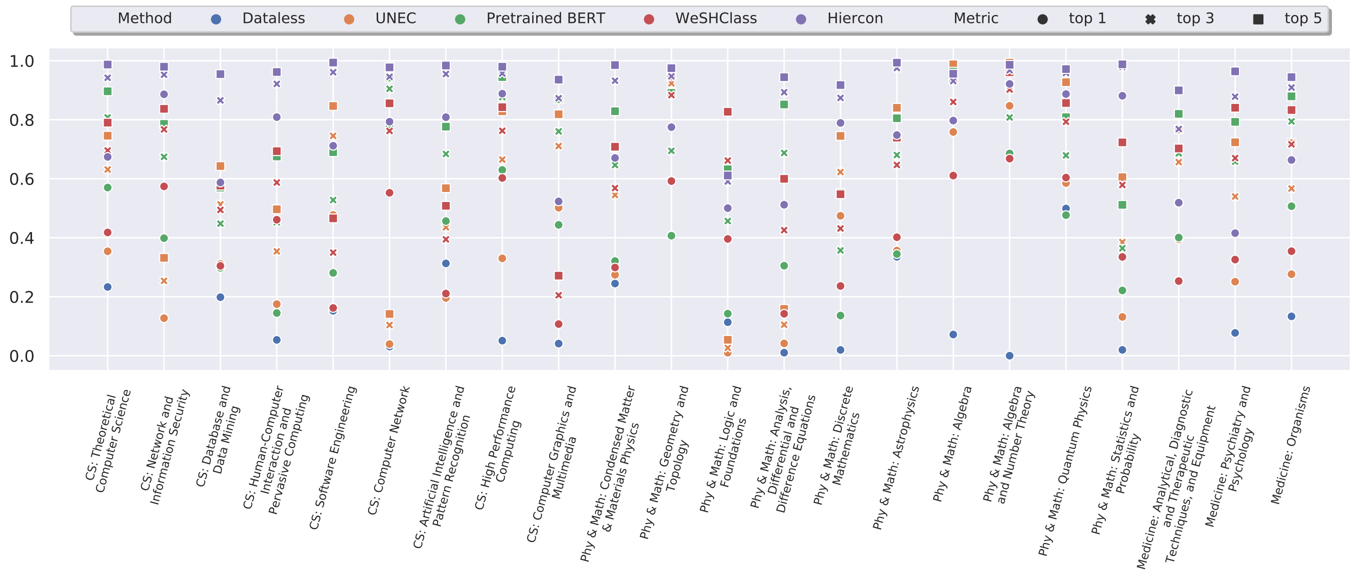


Fig. 5: Detailed Performance evaluation divided by the subject categories. The flat classification accuracy for each sub-fields are recorded top-K prediction score are displayed along specific vertical axis.

representing documents and categories. The concept embedding vector used for similarity computation is trained using the Skip-gram objective as a 50 dimensional embedding vector for each concept. And in the dynamic bin pooling stage, we divide the concept similarities into $K = 16$ bins, 15 of which equally spaced bins between $[-0.5, 1)$ plus 1 for exact match, to be used in bin pooling for the similarity aggregation.

4) Evaluation methodology: In order to evaluate the above methods in a weakly supervised setting, we randomly select 3 documents for each class label and combine them together as the training set, following previous work [20], and we use all the remaining documents as the test set.

Because the category label from each class may contain noise, and that each document may be associated with multiple labels, we adopt the top $k = 1, 3, 5$ accuracy measure [44] to evaluate the performance of each method.

In order to further incorporate tree structure into the evaluation and account for the fact that the prediction into nodes which are closer to the ground truth nodes are less "wrong", we propose a *tree* version of top k accuracy, in addition to the classical *flat* one. Its calculation is based on the following procedure: instead of returning 1 only when the prediction is the same as the ground truth and 0 otherwise, we also return 1 if the prediction is the sibling or the parent node of the ground truth node, and 0 if it is in other part of the tree.

B. Overall Performance with Automatic Evaluation

Overall performance We present the results of the overall performance evaluation in Table II. Hiercon achieves the best performance overall by effectively deriving concept representation for documents and hierarchy labels, and comprehensively utilizing all the concept similarity, and allowing them to flow to downstream relevance computation. Pretrained BERT also works relatively well, which confirms the validity of training label, and the generalizability of pre-trained weights; WeSHClass is able to improve it by efficiently utilizing training signals. Dataless is able to achieve coarse level categorization, as seen by the tree accuracy, but in general suffers bad performance when the knowledge base coverage is low.

Performance evaluation for each category We show in Figure 5 the detailed performance for more specific categories.

	Concepts	
	discriminative	indiscriminative
Computer Science	*fpga (Hardware)* *nash equilibria (Game Theory)* *attacks (Security)*	*framework* *goal* *technique*
Physics Maths	*entanglement (Quantum Physics)* *Jupyter (Planet astrophysics)* *complete graph (Combinatorics)*	*correspondance* *metric* *series*
Medicine	*MRI (Diagnosis)* *treatment (Therapeutics)* *HIV (Viruses)*	*levels* *ability* *regression*

TABLE III: Example discriminative vs indiscriminative concepts discovered in each dataset

For better illustration, we group these them into sub-categories for each subject field. We can observe that although the performance of each method varies by category depending on the difficulty of the corresponding science field, the relative trend stays similar: Hiercon is almost always outperform other baseline approaches, with its top 3 prediction covering the right class most of the time, followed by other methods such as WeSHClass. There are very few cases where Hiercon doesn't perform well, for example, the category "Logic" in the Physics and Mathematics dataset, possibly due to less accurate concept representation for that category. However, in practice we can remedy this by associating categories with more accurate concepts to describe the content of that class.

Performance evaluation with varying numbers of training examples To further investigate how well our model utlize the training data, we perform an ablation study over the number of training instances given to the model. The results are shown in Figure 6, where we give the model various number of training examples and record its top 1 accuracy relative to the original model with the complete training set, as its relative performance measure. We can clearly see that our approach can efficiently fine-tune the model weights with very little number of examples, with 10 training examples, it already reaches a considerable level of accuracy; with 20-30 training examples, the model can approximately recover performance of the original model trained on the full training set.

C. Qualitative Study

Discriminative & indiscriminative concepts If we treat each concept as a document and perform classification on it, we can obtain the direct relevance between concepts and taxonomy nodes. Table III shows some of the most discriminative & indiscriminative concepts. We can see that the concept representation of taxonomy nodes is able to capture latent semantics and make meaningful distinctions on the concept level.

Error case analysis with concept attention Document's relevance to taxonomy nodes can be viewed as a combination of its individual concepts' relevance, weighted by the attention. Figure 5 utilizes this to perform error cases analysis, where concept's attention-weighted relevance to the top-5 predicted category is visualized. From the figure we can clearly see what concepts and how much they contribute to the final prediction. For example, in Figure 7a concepts such as "relational algebra" and "decomposition trees" confuse the model to associate the document with more theoretic subject such as logic

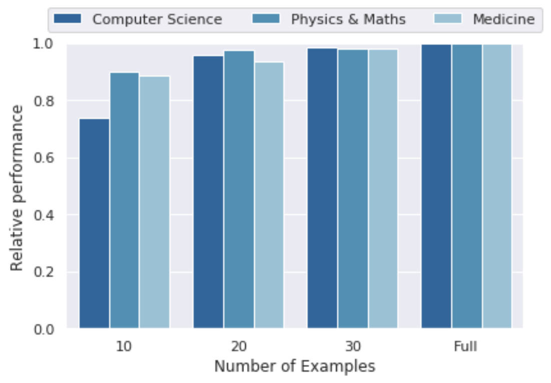
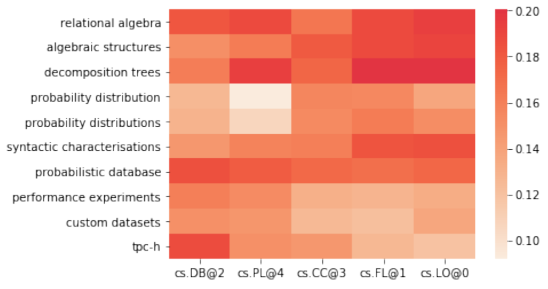
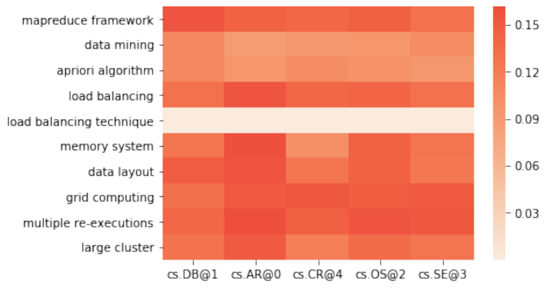


Fig. 6: Ablation study of over the number of training examples. The performance are measured by the relative (top-1) accuracy of the original performance.



(a) Database confused as Logic (Computer Scicence)



(b) Database confused as Hardware & Architecture

Fig. 7: Error case analysis with concept attention. The attention weights distribution over different concepts in a document are illustrated with the colors' color along specific columns.

(coded "cs.LO") and formal languages (coded "cs.FL"). At the meantime, other evidence such as "tpc-h" and "probabilistic database" support the model to partially correctly predict it as database. Similarly, in Figure 7b concepts such as "grid computing" and "load balancing" strongly support the model to predict the document as "Hardware & Architecture" (coded "cs.AR").

VII. Conclusion

In this work, we studied the problem of hierarchical organization of technical documents, where given a set of documents and a set of labels organized as a taxonomy tree, the goal is to classify each document into the taxonomy tree using very few training examples. We proposed a novel concept based framework that learns to represent documents and hierarchy nodes using concepts mined from the corpus, and obtain their relevance by aggregating similarity of individual concepts similarities. We extensively evaluated the proposed approach with state of the art baseline methods, and demonstrated its effectiveness in a wide range of technical domains. As one of several promising future directions, we are planning to dynamically generate the target taxonomy that might better reflect the inner structure of the input corpus. Another direction is to further study how to incorporate the concept semantics to more general text mining and analytics tasks.

References

[1] R. Johnson, A. Watkinson, and M. Mabe, "The stm report."

[2] G. Murphy, *The big book of concepts*. MIT press, 2004.

[3] K. Lamberts, *Knowledge Concepts and Categories*. Psychology Press, 2013.

[4] B. S. Wynar, A. G. Taylor, and J. Osborn, *Introduction to cataloging and classification*. Libraries Unlimited Englewood, CO, 1992.

[5] J. S. Wilkins, "What is systematics and what is taxonomy," *Google Scholar*, 2011.

[6] S. Gopal and Y. Yang, "Recursive regularization for large-scale classification with hierarchical and graphical dependencies," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2013, pp. 257–265.

[7] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 27.

[8] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[9] K. Li, H. Zha, Y. Su, and X. Yan, "Unsupervised neural categorization for scientific publications," in *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 2018, pp. 37–45.

[10] Y. Song and D. Roth, "On dataless hierarchical text classification." in *AAAI*, vol. 7, 2014.

[11] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, 2009, pp. 1003–1011.

[12] L. Rokach and O. Maimon, "Clustering methods," in *Data mining and knowledge discovery handbook*. Springer, 2005, pp. 321–352.

[13] C. Carpineto and G. Romano, "A survey of automatic query expansion in information retrieval," *ACM Computing Surveys (CSUR)*, vol. 44, no. 1, p. 1, 2012.

[14] K. Li, H. Zha, Y. Su, and X. Yan, "Concept mining via embedding," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 267–276.

[15] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[16] J. Shang, J. Liu, M. Jiang, X. Ren, C. R. Voss, and J. Han, "Automated phrase mining from massive text corpora," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 10, pp. 1825–1837, 2018.

[17] T. D. Nguyen and M.-Y. Kan, "Keyphrase extraction in scientific publications," in *International conference on Asian digital libraries*. Springer, 2007, pp. 317–326.

[18] A. McCallum, R. Rosenfeld, T. M. Mitchell, and A. Y. Ng, "Improving text classification by shrinkage in a hierarchy of classes." in *ICML*, vol. 98, 1998, pp. 359–367.

[19] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma, "Support vector machines classification with a very large-scale taxonomy," *Acm Sigkdd Explorations Newsletter*, vol. 7, no. 1, pp. 36–43, 2005.

[20] Y. Meng, J. Shen, C. Zhang, and J. Han, "Weakly-supervised hierarchical text classification," *arXiv preprint arXiv:1812.11270*, 2018.

[21] C. Xiong and J. Callan, "Query expansion with freebase," in *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*. ACM, 2015, pp. 111–120.

[22] J. Dalton, L. Dietz, and J. Allan, "Entity query feature expansion using knowledge base links," in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014, pp. 365–374.

[23] J. Shen, J. Xiao, X. He, J. Shang, S. Sinha, and J. Han, "Entity set search of scientific literature: An unsupervised ranking approach," *arXiv preprint arXiv:1804.10877*, 2018.

[24] C. Xiong and J. Callan, "Esdrank: Connecting query and documents through external semi-structured data," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 951–960.

[25] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2017, pp. 1271–1279.

[26] C. Xiong, J. Callan, and T.-Y. Liu, "Word-entity duet representations for document ranking," *arXiv preprint arXiv:1706.06636*, 2017.

[27] Z. Zhang and V. Saligrama, "Learning joint feature adaptation for zero-shot recognition," *arXiv preprint arXiv:1611.07593*, 2016.

[28] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 2927–2936.

[29] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2014.

[30] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, "Zero-shot learning through cross-modal transfer," in *Advances in neural information processing systems*, 2013, pp. 935–943.

[31] Z. Zhang and V. Saligrama, "Zero-shot learning via joint latent similarity embedding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 6034–6042.

[32] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5327–5336.

[33] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[34] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[35] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," 2016.

[36] D. Q. Nguyen, "An overview of embedding models of entities and relationships for knowledge base completion," *arXiv preprint arXiv:1703.08098*, 2017.

[37] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[38] D. Carmel, M.-W. Chang, E. Gabrilovich, B.-J. P. Hsu, and K. Wang, "Erd'14: entity recognition and disambiguation challenge," in *ACM SIGIR Forum*, vol. 48, no. 2. ACM, 2014, pp. 63–77.

[39] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning-the good, the bad and the ugly," *arXiv preprint arXiv:1703.04394*, 2017.

[40] J. Guo, Y. Fan, Q. Ai, and W. B. Croft, "A deep relevance matching model for ad-hoc retrieval," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 2016, pp. 55–64.

[41] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2333–2338.

[42] O. Egozi, S. Markovitch, and E. Gabrilovich, "Concept-based information retrieval using explicit semantic analysis," *ACM Transactions on Information Systems (TOIS)*, vol. 29, no. 2, p. 8, 2011.

[43] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[44] M. Lapin, M. Hein, and B. Schiele, “Top-k multiclass svm,” in *Advances in Neural Information Processing Systems*, 2015, pp. 325–333.