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†
The Ohio State University¶
{klee, shiyangli,syavuz,hanwen_zha,xyan}@cs.ucsb.edu su.809@osu.edu

***Abstract*—In this work we study the hierarchical organization of technical documents, where given a set of documents and a hierarchy of categories, the goal is to assign documents to their corresponding categories. Unlike prior work on supervised hierarchical document categorization that relies on large amount of labeled training data, which is expensive to obtain in closed technical domain and tends to stale as new knowledge emerges, we study this problem in a weak supervision setting, by leveraging semantic information from concepts. The core idea is to project both documents and categories into a common *concept embedding space*, where their fine-grained similarity can be easily and effectively computed. Experiments over real-world datasets from the subject of computer science, physics & mathematics, and medicine demonstrated the superior performance of our approach over a wide range of state of the art baseline approaches.**

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. Advanced techniques for better understanding and organizing the scientific literature are therefore in great demand. According to cognitive science studies [2], [3], [4], a key management strategy for such information is to organize them into a hierarchy of categories, which has been widely used in various domains such as library science [5], internet directories[1] [2], patents[3], among many others.

Traditional methods for this problem mostly focus on a supervised setting [6], [7], [8], which relies on a significant number of manually labeled document-class pairs and is in general inapplicable to our setting, because of the following two reasons: 1) manual labeling requires strong domain expertise and thus is very expensive to obtain 2) the set of categories can become obsolete quickly as the domain evolves and new knowledge emerges [9]. Yet another line of research [10] exploits distant supervision signal from external knowledge bases [11], which are in general not available in many fine-grained, closed technical domains.

We propose to study the problem of hierarchical document categorization under a *weak supervision* setting. More specifically, given a set of documents and a category hierarchy, the goal is to assign documents into the different categories based on very few number of labeled documents. [4], where only a few labeled documents are provided. The hierarchical nature of the problem makes the problem even more challenging: the weakly supervised classifier need to jointly consider the large number of category labels as well as their relations between each other.

There are a number of possible baselines to consider. One may use a clustering approach, which detects coherent sets of documents in an unsupervised fashion. The hierarchical structure can be incorporated into the clustering process by generating hierarchies along with the clustering [12], or by employing flat clustering approaches and map clusters to the leaf nodes of the hierarchy. 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 adapt flat unsupervised categorization models [9] to the hierarchical setting, e.g. by following the top-down or bottom-up paradigm [12].

We explore a vastly different categorization paradigm. It is based on the observation that technical documents are typically organized around *concepts* [14], [15], 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 model, HierCon, which leverages concepts in technical documents to form semantic representations for both the documents and hierarchical categories and infer their associations. Unlike previous approaches that rely on *model specific* engineering to incorporate hierarchical structure, for example, by imposing similarity or orthogonality constraints between parent and children classifiers [6], [16], we propose to represent the categories as *distributions* over concepts, which allows for more flexible combinations of the semantics of neighboring nodes in the hierarchy.

Concept mining [14] can be leveraged to discover concepts and their occurrence locations in the documents, and learn

[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 terminologies "organize", "categorize", "classify" and the set of terminologies "hierarchical label", "taxonomy node", "hierarchical category" interchangeably depending on the context.

2374-8486/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDM.2019.00048

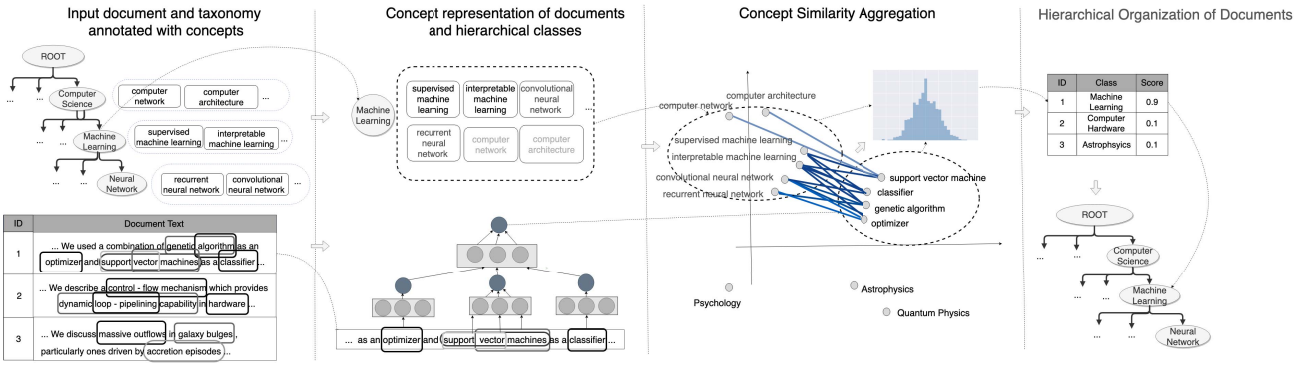


Fig. 1: HierCon Framework Overview.

their vectorized semantic embedding representation without supervision signal. For example, given a text piece "support vector machine", one may recognize either "support vector machine" as a concept, or other alternatives such as "support vector" as a concept. The task now becomes, to select concepts that are 1) most valid among different alternatives, and 2) most important to the document's main topic. To that end, we propose a novel, adaptive concept level document representation model based on the hierarchical neural attention mechanism [17], which models the validity and importance of the concept recognition as a natural hierarchical process, and dynamically adjust the concept representation and the associated model weights based on downstream performance. Documents and categories are thus projected into a common *concept embedding space*, as a weight distribution over concepts along with the vectorized, semantic embedding associated with each concept.

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 observation is that, many concepts in the document bear strong, discriminative information about the documents' categories, via their associated concepts. Motivated by this, we propose a principled approach for aggregating all possible concept interactions between the documents and each of the possible categories, to comparatively obtain document-category relevance and perform categorization.

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 state of the art baseline methods under various performance metrics. 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 categories and the documents using concepts mined from the unlabeled documents and project them into a common *concept embedding space*, from which we obtain direct relevance signal and assign documents to categories.
- 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 follows a supervised approach. Hierarchical information can be taken into account by either enforcing similarity or dissimilarity constraints into the classification model [18], [19], [6], [7], or by modifying the training data and prediction results to enforce hierarchy consistency [20]. On the other hand, distant supervision based methods such as dataless hierarchical classification [10] employ large knowledge bases such as Wikipedia to generate semantic representations of both documents and labels, and associate them together via a top-down or bottom-up scheme. These approaches either heavily rely on the availability and quality of labeled training set for specific classes, or expect the corpus to be covered by an external knowledge base, which are inapplicable to our setting. A recent work by Meng et.al [21] studied a very similar problem, where they assume a set of labeled documents *for each category*, and leverage these documents to represent the category semantics and perform categorization following a bootstrap procedure. Such an approach will be more sensitive to the training data: one noisy labeled document for a category could lead to much bigger error in the further bootstrapping stages.

B. Concept/entity aware representation

Another line of related work is on concept/entity aware representation, where they utilize information from a knowledge base, such as entity description, entity type or even links to other entities to improve different stages in the

information retrieval pipeline such as query expansion [22], [23] and ranking [24]. Follow up work directly makes use of the annotated entities in the documents as its semantic representation [25], [26], [27], [28], to derive insights about the documents and serve downstream applications. The above approaches provide evidence for the key roles of the concepts in representing texts, and supports our further exploration of hierarchical document organization based on concepts.

C. Classification without explicitly labeled training data

Yet another line of related work can be categorized as performing classification without explicitly labeled training data. The common characteristics of these approaches are to exploit the extra information contained in the label. A highly related approach is zero-shot learning from the computer vision domain, which deals with the problem of categorizing against new unseen categories. These approaches mostly follow a representation learning based approach, which relates unseen classes and seen ones by extracting features of each class [29] and/or inputs [30], [31], [32] and learn a vector representation, in order to generalize training data to those new labels. In text domain, neural embedding based technique [33], [34], [35] has been the main workhorse to extract information from unsupervised data and associate semantics with labels. Li et.al [9] provides an initial study along this direction, where they exploit *concepts* to best learn the embedding and perform unsupervised categorization.

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 $d \in \mathcal{D}$ being a sequence of words in the form of $w_1 w_2 \dots w_{|d|}$. In addition, we're given a set of target categories $\mathcal{Y}$, organized into a tree structure $\mathcal{T}$, where of all the categories $\mathcal{Y}$ are nodes in the tree, and each category $y \in \mathcal{Y}$ is associated with one parent $P(y) \in \mathcal{Y}$, a set of ancestors $\mathcal{A}(y)$, e.g. its parent, grandparent, and so forth. We study this problem in a weakly supervised setting, where users are allowed to provide a set of labeled training data, as a set of $l \ll \mathcal{D}$ document-label pairs. The end goal of hierarchical organization is, then, to associate each document $d \in \mathcal{D}$ with one or more relevant labels $\mathcal{L}(d) \subseteq \mathcal{Y}$.

Our proposed approach is illustrated in Figure 1. In order to leverage *concepts* to represent documents and categories, we first perform concept mining [14], i.e. identifying meaning bearing units such as "support vector machine" and "generative adversarial network" from text. Specifically, the output will be 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 section IV, we will generate the concept level representation $\varphi : \mathcal{D} \to \mathbb{R}^{|\mathcal{V}|}$, as a function that maps each document $d \in \mathcal{D}$ to a $\mathcal{V}$ dimensional vector, where each dimension denotes the weighted association to each specific concept.

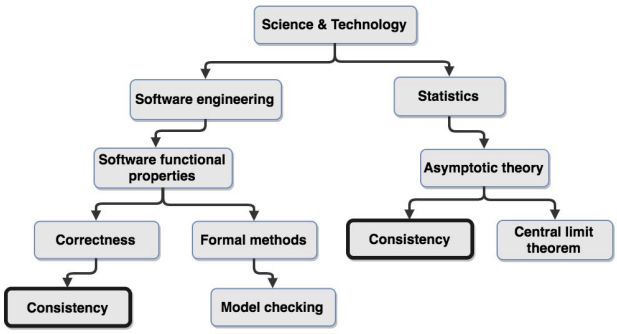


Fig. 2: An example hierarchy of categories showcasing the importance of *path semantics*. The category node "Consistency" under Software engineering and another node "Consistency" concept under the Statistics can be distinguished by their ancestors, descendents and neighbors.

By associating target categories with relevant concepts in the concept vocabulary $\mathcal{V}$ based on string similarity, we can derive the *basic semantics* for each category [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 thus will have high value in the corresponding dimensions in its concept representation. However, solely relying on *basic semantics* may miss important hierarchy information and lead to ambiguity, as illustrated by the example below.

Example 1 (basic semantics of category) Consider the category hierarchy which we want to categorize documents into as shown in Figure 2. The hierarchy is artificial but manifests the following key observations: (1) Many interesting and concrete concepts about a category node are buried deep in its descendent, which is not directly recovered by the name of the category itself. For example, descendent nodes such as "Asymptotic theory" and "Central limit theorem" provide much richer information to the ancestor category, "Statistics", compared to simply relying on its literal name. (2) Conversely, the location of a category node in the hierarhcy, e.g. what 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 are two completely different categories which share little overlap with each other. Without the hierarchy context, it will be difficult to distinguish between these two.

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

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

where we use $C(y) = P^{(-1)}(y)$ to denote the set of children

of node y, and avg to denote the aggregation function for averaging the value, which in this work we instantiate as the element-wise arithmetic mean.

Motivated by the second observation, we introduce *path semantics* $\phi^{(p)}(v)$ for each node $y \in \mathcal{Y}$ to take into account the location of category node in the hierarchy in determining its semantics, as an averaging over the *aggregated semantics* of the current node y and its ancestors $\mathcal{A}(y)$. [5] The path semantics $\phi^{(p)}(v)$ will then be used as the final representation for each category node y, $\phi(y)$. Specifically,

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

Given the concept representation of category nodes $\phi(\cdot)$ and documents $\varphi(\cdot)$, the task is then 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 rewards following the reinforcement learning paradigm [36]. 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 the *big-bang approach* [8], that classifies each document against all nodes at the same time, and predict the label as the one that maximizes the similarity. Specifically, for model inference, we obtain the prediction $\hat{y}(d)$ for each document d by

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

Here 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 logits.

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 recursively obtains the *aggregated semantics* of category nodes using its children nodes. These are followed by a top-down pass, that computes the *path semantics* $\varphi_v^{(p)}$ for each label. Based on the semantic representations of labels and documents, a relevance function $\mathcal{S}(\phi(y), \varphi_d^{(p)}$ 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.

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. Furthermore, it allows for more flexible categorization decision between leaf nodes and internal nodes: a document is assigned to leaf if it is more similar to a specific sub-field; if it is equally similar to them, it should stay at a more general level and get assigned to their parent node. Last but not least, it is interpretable and naturally provides not only the most likely category, but also the top K category to be served as additional information for each document.

[5] We overload the definition of avg when called with two arguments $avg(\cdot, \cdot)$, to make it return the average over each argument. in other word, the second argument, in our case the *aggregated semantis* of the current node, will take exactly a 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 hierarchy $\mathcal{T}$, number of labels to predict for each document K
Output: : top K labels $\mathcal{L}(d)$ for each document $d \in \mathcal{D}$
Perform concept mining to obtain concept vocabulary $\mathcal{V}$, concept representation $\{\vec{\varphi(d)}|d \in \mathcal{D}\}$, and a set of concept embedding vectors $\{\theta_v|v \in \mathcal{V}\}$ from raw corpus $\mathcal{D}$
for node y in the hierachy $\mathcal{T}$ **do**
  Obtain *basic semantics* $\phi^{(b)}(y) \in \mathbb{R}^{|\mathcal{V}|}$ based on the concept vocabulary $\mathcal{V}$ and the category node y
end for
for level of tree $T \in \mathcal{T}$, bottom up **do**
  for node y in the level of tree T **do**
    Compute *aggregate semantics* $\phi^{(a)}(y) \in \mathbb{R}^{|\mathcal{V}|}$ according to Eq. (2).
  end for
end for
for level of tree $T \in \mathcal{T}$, top down **do**
  for node y in the level of tree T **do**
    Compute the final representation as the *path semantics* for category node y, as $\phi(y) \in \mathbb{R}^{|\mathcal{V}|}$ according to Eq. (3).
  end for
end for
Compute relevance scoring function $\mathcal{S}(\varphi(d), \phi(y))$ for each $d \in \mathcal{D}$ and node y in the hierarchy $\mathcal{T}$ based on the computed representation $\varphi(\cdot), \phi^{(\cdot)}$ and concept embedding vectors $\{\theta_v|v \in \mathcal{V}\}$.
return top K labels $c \in \mathcal{C}$ for each document $d \in \mathcal{D}$ that has the highest relevance scores

IV. Concept representation for documents

In this section we discuss our approach for obtaining the concept representation $\varphi(d)$ of each document $d \in \mathcal{D}$. We follow a two-step approach: We start with a candidate generation stage where we follow [14] to obtain 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, where both "vector machine" and "support vector machine" are recognized as concept candidate. The goal of the next stage, candidate selection, is then to select concepts like "support vector machine" as the correct concept occurrence, and furthermore among the correct ones select the important

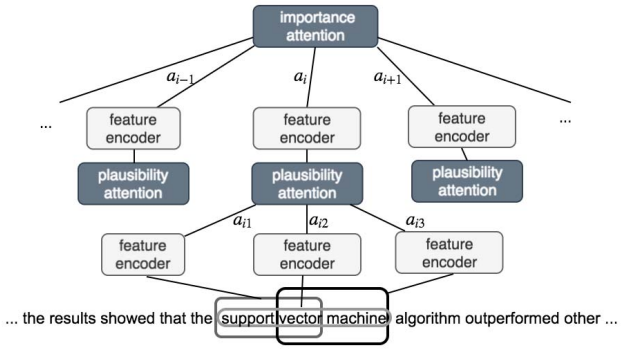


Fig. 3: Illustration of the hierarchical attention mechanism.

concepts in the document to form the document's concept representation.

Specifically, we follow the assumption from the entity recognition literature [37] and require that the resulted textual occurrences of these concepts after the candidate selection stage to not overlap with each other. More formally, given a document d as a sequence of words, the outcome of candidate generation will be a sequence of *super-concepts* [14], $u_1, u_2, u_3, \ldots, u_l$, l being the length of such sequence, where each u_i contains a set of overlapping candidate concepts $c_{i1}, c_{i2}, \ldots, c_{i|u_i|}$, $|u_i|$ being the number of candidates in the *super-concept*, and each u_i does not overlap with other *super-concepts*. The goal of candidate selection, then, is to disentangle the overlapping candidate concepts and select the valid concept for each *super-concept* u_i, and then to determine the importance among the selected concepts from each *super-concept* and generate the concept representation of documents as a weighted distribution across concept vocabulary $\varphi(d) \in \mathbb{R}^{|\mathcal{V}|}$.

We propose an adaptive concept representation architecture based on the hierarchical attention mechanism [17], which directly combines noisy concept candidates into the final document representation in an end-to-end fashion, simultaneously performing the task of selecting valid concepts and the task of selecting important ones. As shown in Figure 3, in the first level of attention, inside each *super-concept*, the model will choose to attend to one of its most probable candidates; in the second level, a document-wise attention is drawn to select the most important concepts from the selected valid concept from each super-concepts.

Specifically, the attention score for the tth candidate in the ith superconcept, denoted a_{it}, for each $1 \leq t \leq |u_i|, 1 \leq i \leq l$, can be obtained as follows.

$$a_{it}^0 = f_{it} \cdot W_0 \tag{4}$$

$$a_{it} = \frac{exp(a_{it}^0)}{\sum_t (exp(a_{it}^0))} \tag{5}$$

Here W_0 is the model parameters for weighting different feasibility features, and a_{it}^0 are used to denoted intermediate output in the network. f_{it} referes to the feature vector associated with the the tth candidate in the ith superconcept, which is obtained based on the grammatical features, occurrence frequency, statistical significance and context based features such as the ones used in Autophrase [38] and ECON [14]

At the second level of attention, where the goal is to capture concept importance, the attention score for the selected concepts from the ith super-concept, denoted a_i, can be obtained as follows.

$$a_i^0 = f_i \cdot W_1 \tag{6}$$

$$a_i = \frac{exp(a_i^0)}{\sum_t (exp(a_i^0))} \tag{7}$$

Here W_i is the weight parameters, a_i^0 is the intermediate model output. f_i refers to the feature vector associated with the selection concepts from the ith super-concept, and is obtained based on features such as occurrence location, length of the concept, which logical section the concept occurs in [39]. The details of the feature used can be found at the full version of the paper [7].

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, if we index the $|\mathcal{V}|$ dimensional vector $\varphi(d)$ using each specific concept $c \in \mathcal{V}$, then the value of $\varphi(d)$ along that index, denoted $(\varphi(d))_c$, can be computed based on the two level attention, as

$$(\varphi(d))_c = \sum_{1 \leq i \leq l} a_i \cdot \sum_{1 \leq t \leq |u_i|} (a_{it} \cdot \mathbb{I}(c_{ij} = c)) \tag{8}$$

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 category and document $(y, d) \in \mathcal{Y} \times \mathcal{D}$ based on their concept representations.

The approach of measuring 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 [40]; Information retrieval extracts features from documents and queries, and build a model to predict their relevance [41], [42]. These models are still supervised in nature which requires standard training data for a subset of labels ("seen" classes). There are also models that leverage an external knowledge base to derive vector representation and relevance [10]. None of the above are applicable to our scenario, where both training examples and knowledge base coverage are scarce.

To address this, we leverage the fact that the documents and category labels are projected into the same *concept embedding space*, and obtain direct relevance signal based on their aggregated distance in such space. We propose a novel *similarity aggregation* framework that exploits the innate similarities between concepts based on their semantic embedding, comprehensively considers all possible interactions between concepts in the documents and category labels, and

[7] https://sites.cs.ucsb.edu/~klee/papers/ICDM19_HierCon.pdf

dynamically *learns* the relevance signal as a function of the innate similarities between concepts to flexibly aggregate them together.

Formally, we assume we're given the concept-wise similarity function $S(c_i, c_j)$ for all possible pairs of concepts $c_i, c_j \in \mathcal{V}$, which in this work we instantiate as cosine similarity. Our goal is, then, to aggregate the set of concept similarities for each document d and category label y, based on their concept weight distribution $\varphi(d)$ and $\phi(y)$. The problem is that, the individual similarity strengths, e.g. as measured by cosine similarity, may not directly correspond to their contribution in the document-category level relevance. How can we more flexibly understand the concept similarity strength, while at the same time respect their weights, as indicated by the document and category's concept representation?

We leverage the techniques of bin-pooling [41], and first discretizes the similarities into K bins, with the kth bin counting the number of values between the range $[st_k, end_k)$, $1 \leq k \leq K$. As a result, the similarities of each pair of concept $(c_i, c_j) \in \mathcal{V} \times \mathcal{V}$ will be mapped to a specific bin, denoted as $bin(S(c_i, c_j))$. Next, we *embed* each bin into an (arbitrarily) learned representation using the embedding parameters $\omega \in \mathbb{R}^K$, whose k-th dimension describes the embedding signal for the k-th bin. As a result, each concept pair (c_i, c_j) can then be mapped to a representation $M(c_i, c_j|\omega)$ which directly corresponds to its contribution to the final relevance score:

$$M(c_i, c_j|\omega) \triangleq \omega_{bin(S(c_i,c_j))} \qquad \forall (c_i, c_j) \in \mathcal{V} \times \mathcal{V} \tag{9}$$

Given the concept pairs $M(c_i, c_j|\omega)$, we directly weigh it with the document's weight towards the specific concept c_i, $(\varphi(d))_{(c_i)}$, as well as the category' weight towards concept c_j, $(\phi(y))_{c_j}$, and obtain the final relevance as

$$\mathcal{S}(\varphi(d), \phi(y)|\omega) \triangleq \sum_{(c_i,c_j)\in\mathcal{V}\times\mathcal{V}} (\varphi(d))_{c_i} \cdot (\phi(y))_{c_j} \cdot M(c_i, c_j|\omega) \tag{10}$$

Intuitively, the above model can be viewed as a process of *dynamically* generating a matching histogram between document and labels based on learn-able weight distribution for both categories and documents, and then map the raw matching histogram to the final relevance score through a learned function. We therefore denote this model as *dynamic bin pooling*. The process of dynamically generating the matching histogram is fully differentiable and therefore allows us to fine-tune both the document and category's concept representation as well as the mapping from the raw matching histogram to the final relevance score, in an end-to-end manner.

Gradient path saving Directly optimizing the similarity aggregation model via current hardware may be hard due to the large number of possible concepts and their interactions. For example, it can be shown that there are at least $\Omega(K|\mathcal{Y}||\mathcal{V}|^2)$ gradients that need to be stored and back-propagated which can easily exceed the capacity of modern GPU; even if we store the concepts as a sparse matrix to only keep the non-zero entries, the set of concepts in the category's concept representation could still be large due to *path semantics* computation, and we need to consider the relevance for all y in $\mathcal{Y}$ for both training and prediction.

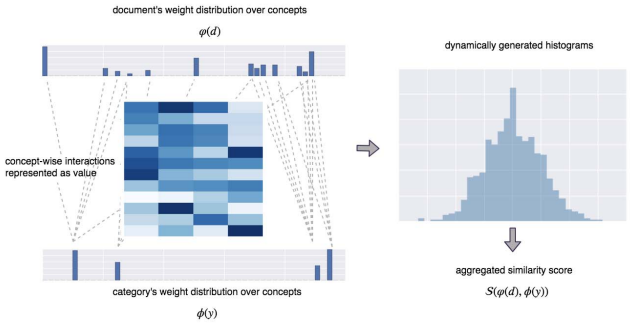


Fig. 4: Dynamic bin-pooling for similarity aggregation.

To address this, we propose the *gradient path saving* approach, which explores the cheapest path for back-propagating the gradients based on the re-parameterization trick. Specifically, it can be shown that the gradients associated with the bin-weight ω can be obtained by

$$\frac{\partial \mathcal{S}(\varphi(d), \phi(y)|\omega)}{\partial \omega_k} = (\varphi(d) \otimes \phi(y)) \odot M_k \quad 1 \leq k \leq K \tag{11}$$

where $\otimes$ denotes vector outer-product, $\odot$ denote element-wise sum, and each entry in the matrix M_k stores the look-up results of corresponding elements in M for the kth bin, as

$$M_k(c_i, c_j|\omega) = \mathcal{I}(bin(S(c_i, c_j)) = k) \tag{12}$$

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, we can show the gradient for the document's concept representation and the category's concept representation 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 the values "scattered around", where bins with lower similarity strength have even higher weight than those with higher similarity strength, which is against our design. To ensure that the bin weights have meaningful values, we follow a *monotonicity enforcement* approach, that leverages re-parameterization trick to impose monotonicity constraint for the bin-weights. Specifically, instead of storing the raw bin weights ω_k for $1 \leq k \leq K$, we store the *differential* of bin weights, denoted ω'_k. And the kth bin-weight ω_k, now a computed value, will be obtained by

$$\omega_k = \sum_{1\leq j\leq k} \text{ReLU}\ (\omega'_j) \qquad \forall k, 1 \leq k \leq K \tag{13}$$

where ReLU refers to the relu activation function[8] which enforces monotonicity. In other words, only the positive weight differences will be counted, and the network is constrained to

[8]https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

TABLE I: Dataset Statistics

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

either learn to increase the bin-weights with higher similarity strength, or to keep it at the same level as the bin-weights with lower similarity strength.

Since the parameters in the *similarity aggregation* model are only associated with concepts as well as their similarity strength, and that each labeled document-category pair will correspond to a much larger number of concepts that we train these parameters on, we can effectively tune these parameters using much smaller number of labeled training examples. In addition, because it trained on the concept level information and not limited to knowledge about specific classes, it also naturally extends to zero-shot setting and make predictions about unseen classes labels based on their concept representation.

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", with category hierarchy obtained by aligning arXiv's category label [9] with aminer.org's academic conference classification [10].

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[11] and the physics subject headings[12].

Medicine The Medicine corpus is obtained by crawling abstracts from Pubmed[13], and retaining documents associated with the top 3 level of the Medicine subject headings (Mesh) [14] under the top "Organisms", "Analytical, Diagnostic and Therapeutic Techniques, and Equipment" and "Psychiatry and Psychology".

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

WeSHClass [21] 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[15] and use their recommended settings.

Dataless refers to the hierarchical dataless classifcation approach [10], which is a state-of-the-art distant supervision based method in the hierarchical classification literature. It works by first obtaining vector representations of documents and class labels via its similarity with Wikipedia articles [43], and associate documents with class labels based on their vector representations. We have follow the original dataless classification implementation from the authors[16], and then apply it with a bottom-up scheme.

Pretrain BERT is currently the state-of-the-art deep learning approach for downstream NLP tasks, which leverage deep self-attention network based architecture with weights pretrained on large training corpus. Specifically, we add a classification layer upon the sentence representation from the last transformer layer, similar to the setup for the BERT in GLUE classification tasks [44], and fine-tune all the layers and report the performance 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 learning a concept embedding graph, where similarity to categories are propagated. We follow the personalized page rank approach[17] to propagate the similarity on the concept graph due to its stability for larger number of categories.

refers to our proposed approach. We utilize the concept mining technique ECON [14] to obtain concepts for 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. And in the dynamic bin pooling stage, we divide the concept similarities into $K = 16$ bins, 15 of which are equally spaced bins between $[-0.5, 1)$ with an addition bin counting for exact matches.

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, similar to previous work [21], and use all the remaining documents as the test set. Because the category label from each class may contain noise, and that each document

[9]https://arxiv.org/corr/subjectclasses

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

[11]https://www.maa.org/press/periodicals/loci/joma/subject-taxonomy

[12]https://physh.aps.org/

[13]https://www.ncbi.nlm.nih.gov/pubmed/

[14]https://www.nlm.nih.gov/mesh/meshhome.html

[15]https://github.com/yumeng5/WeSHClass/

[16]https://cogcomp.org/page/download_view/Descartes

[17]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 category nodes are recorded and compared against the ground truth according to both flat classification accuracy and tree based accuracy

		WeSHClass		Pretrain BERT		Hier-SVM		Dataless		UNEC			
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.

may be associated with multiple labels, we adopt the top $k = 1, 3, 5$ accuracy measure [45] to evaluate the performance of each method.

In order to further incorporate tree structure into the evaluation and to encourage the predicting category nodes that are closer to the ground truth category nodes are less "wrong" than predicting those that are farther away, we follow previous hierarchical classification literature [46] and evaluate our approaches with the *tree* based accuracy. We calculate this using the following procedures: we also return 1 if the prediction is the "direct relative", i.e. the sibling or the parent node of the ground truth node, and 0 otherwise.

B. Overall Performance with Automatic Evaluation

Overall performance We present the results of the overall performance evaluation in Table II. 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 from bad performance when the knowledge base coverage is low.

Performance by subject category We show in Figure 5 the detailed performance for each categories. For better visualization, categories are grouped according to the immediate children categories for each subject field. From the results we can see that although the performance of each method varies by category depending on the difficulty, the relative trend stays similar: almost always outperform other baseline approaches, with its top 3 predictions covering the right class most of the time, followed by other methods such as WeSHClass. There are very few cases where 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, which is remediable in practice as we revise the concept representations and associate more descriptive concepts with that category.

Performance evaluation with varying numbers of training

	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

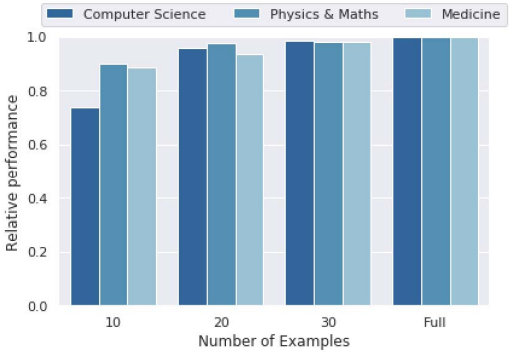


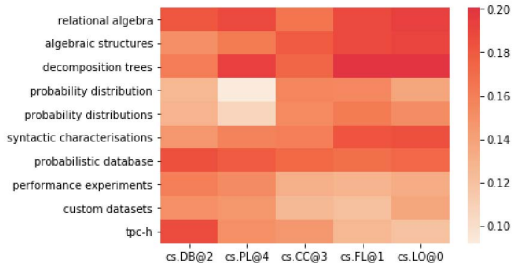
Fig. 6: Ablation study of over the number of training examples. The performance are measured by the relative (top-1) accuracy of the original performance.

examples To further investigate how well our model utilizes the training data, we perform an ablation study over the number of training examples given to the model. The results are shown in Figure 6, where we plot the number of training examples along with the accuracy relative to the corresponding model trained with the complete training set, as its *relative performance measure*. From the results we can clearly see that can efficiently fine-tune the model weights using very little number of examples. With 10 training examples, it already reaches a considerable level of accuracy; with 20-30 training examples, the model is able to almost recover original performance trained with the full training set.

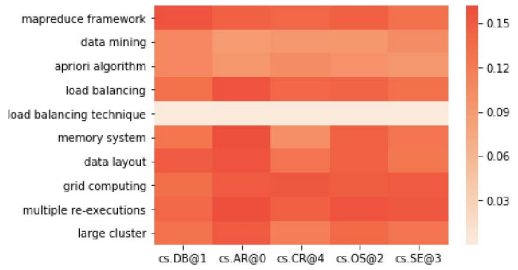
C. Qualitative Study

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

Error case analysis with concept attention The relevance between documents and categories 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



(a) Database confused as Logic (Computer Scicence)



(b) Database confused as Hardware & Architecture

Fig. 7: Error case analysis with concept attention. The matrix of attention weights distribution are illustrated colors of various densities.

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 (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 category organized as a hierarchy, the goal is to assign each document into the categories using very few training examples. We proposed a novel concept based framework that learns to represent documents and hierarchy nodes using the concepts mined from large amount of corpus, and obtain their relevance by aggregating the interactions of individual concepts. We extensively evaluated the proposed approach with state of the art baseline methods, and demonstrated its effectiveness in a wide range of technical domains. One of several promising future directions is to jointly optimize the generation of category hierarchy and hierarchical organization of technical documents. Another direction is to leverage the concept semantics to improve more general text mining and downstream analytical 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] J. S. Wilkins, "What is systematics and what is taxonomy," *Google Scholar*, 2011.

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

[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] K. Li, Y. He, and K. Ganjam, "Discovering enterprise concepts using spreadsheet tables," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017, pp. 1873–1882.

[16] D. Zhou, L. Xiao, and M. Wu, "Hierarchical classification via orthogonal transfer," 2011.

[17] 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.

[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] S. Kiritchenko, S. Matwin, R. Nock, and A. F. Famili, "Learning and evaluation in the presence of class hierarchies: Application to text categorization," in *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 2006, pp. 395–406.

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

[22] 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.

[23] 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.

[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] 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.

[28] K. Li, P. Zhang, H. Liu, H. Zha, and X. Yan, "Poqaa: Text mining and knowledge sharing for scientific publications," 2018.

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

[30] 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.

[31] 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.

[32] 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.

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

[37] 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.

[38] 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.

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

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

[41] 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.

[42] 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.

[43] 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.

[44] 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.

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

[46] A. Kosmopoulos, I. Partalas, E. Gaussier, G. Paliouras, and I. Androutsopoulos, "Evaluation measures for hierarchical classification: a unified view and novel approaches," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 820–865, 2015.