Concept Mining via Embedding

Keqian Li*, Hanwen Zha*, Yu Su†, Xifeng Yan*
*Department of Computer Science, University of California, Santa Barbara, CA, USA
†Department of Computer Science and Engineering, The Ohio State University, OH, USA
{klee, hwzha, xyan}@cs.ucsb.edu, su.809@osu.edu

***Abstract*—In this work, we study the problem of concept mining, which serves as the first step in transforming unstructured text into structured information, and supports downstream analytical tasks such as information extraction, organization, recommendation and search. Previous work mainly relies on statistical signals, existing knowledge bases, or predefined linguistic patterns. In this work, we propose a novel approach that mines concepts based on their occurrence contexts, by learning embedding vector representations that summarize the context information for each possible candidates, and use these embeddings to evaluate the concept's global quality and their fitness to each local context. Experiments over several real-world corpora demonstrate the superior performance of our method. A publicly available implementation is provided at https://github.com/kleeeeea/ECON.**

I. Introduction

The explosive growth of textual data is becoming prohibitive. According to [1], there are about 2.5 million scientific articles published in the year 2014, and this number is still growing at an annual rate of 3%. Advanced techniques for better organization, navigation, and knowledge extraction for texts are in great demand. A first step towards this goal is effective discovery of concepts, i.e., to identify integral units such as "generative adversarial network" and "support vector machine" from raw text. It plays an essential role in transforming unstructured text into structured information [2], [3], constructing knowledge bases [4], [5], [6], [7], and producing meaningful representation of texts to support downstream analytical tasks [8], [9], [10], [11].

One important property of concepts is that some of them may correspond to single words, while the others correspond to multiple word expressions, such as "Folic acid" and "Muscarinic acetylcholine receptor". According to linguistic studies, these idiosyncratic, multi-word expressions should be treated as single units in further analysis, and they are abundant: the number of multi-word expressions is estimated to be "of the same order of magnitude as the number of simplex words in a speaker's lexicon" [12], [13]. In an analysis of more specific, technical terms, [14] points out that multi-word expressions are preferred for specific technical terms because "single words in general vocabulary" are "inherently ambiguous". Perhaps this is the reason driving the research in phrases mining [2], [3] and chunking [15], among many others.

A simple but quite effective approach is to extract word sequences based on predefined grammatical patterns, which is heavily used in areas such as entity/relation type inference, and knowledge construction [4], [16], [5]. However, rule based methods are too simple to capture linguistic variations. On the other extreme, there are end-to-end models that abstract the problem as sequence tagging, and leverage complex models such as deep neural network to learn to capture their occurrences [15]. Unfortunately, we do not always have a large labeled corpus, and even if we do, it will be hard to transfer it to unseen domains: a model that is trained to capture "vector machine" may produce the right results in hardware design, but in the area of machine learning, "support vector machine" should be the correct one. This is especially true when knowledge are constantly evolving and new texts with newly invented concepts keeps emerging.

TABLE I: A hypothetical example of sentence raw frequency.

Text	Frequency
⌈logistic regression⌋ is a supervised machine learning algorithm	10000
we use ⌈logistic regression⌋ for this classification task	5000
our approach outperform ⌈logistic regression⌋ on this dataset	2000
⌈P random⌋ is a supervised machine learning algorithm	100
we use ⌈P random⌋ for this classification task	50
our approach outperform ⌈P random⌋ on this dataset	20
P vs NP	10000
we perform random walk on this graph	10000

A key question is: *Can we utilize an unlabeled corpus to perform concept mining?* Most work in this area relies on statistical significance to estimate concept quality [17], [18]. The state of the art approaches from this angle are SegPhrase and AutoPhrase [2], [3], which rank salient key-phrases by occurrence and co-occurrence frequencies, and based on that re-identify their occurrences in text and rectify these frequencies. These methods work mostly well in identifying frequent concepts in a domain, but would fail to capture *long tail* infrequent concepts where statistical signals are less reliable. This is illustrated in the following example.

Example 1 (Concept Extraction) Consider a corpus consisting of sentences with frequencies shown in Table 1. The numbers are hypothetical but manifest the following key observations: (1) Concepts usually occur under certain usage contexts; (2) Contexts are often shared with concepts with similar meaning; (3) Interesting, but infrequent concepts may be formed by grouping several frequent words together.

In the above example, it will be hard for statistical significance based approaches to capture high quality concepts such as "P random", since its raw frequency and its relative occurrence frequency (with respect to each component words) are low. However, its usage contexts are highly similar to other machine learning algorithms such as "random forest", indicating it's likely a valid concept, instead of two words

978-1-5386-9159-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDM.2018.00042

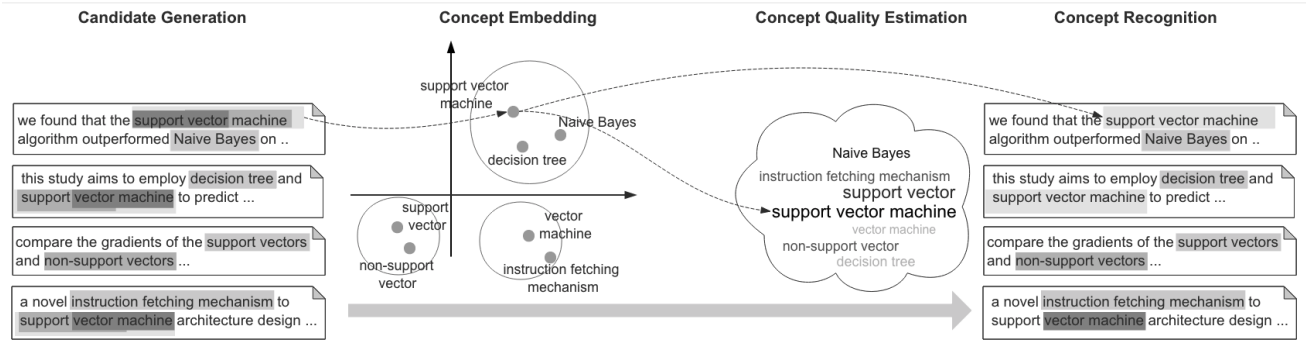


Fig. 1: Pipeline of our concept mining approach

randomly co-occurring together. If we decouple the contexts from the occurrences, we will not be able to find such concept correctly. This becomes a challenging issue for statistical significance based concept mining approaches.

In addition to *concept extraction*, which extracts a set of high quality concepts, contexts can be leveraged to help *concept recognition*, i.e. to determine whether and what concepts occurred in text.

Example 2 (Concept Recognition) Consider the following two sentences

1. We found that the ⌈support vector machine⌋ algorithm outperformed Naive Bayes on this dataset.
2. We use a novel instruction fetch mechanism to support ⌈vector machine⌋ architecture design.

In the first sentence, the concept "support vector machine" should be recognized: The context clearly indicates that it is a machine learning algorithm. In the second one, the context indicates that it is in hardware domain and "vector machine" is more appropriate. Methods such as Autophrase [2] will either recognize them both as "support vector machine", or both as "vector machine". They are not able to change the recognition based on contexts. By leveraging local context clues such as "Naive Bayes" or "instruction fetching logic", and comparing them with common occurrence contexts of "support vector machine", or "vector machine", we can more confidently recognize their occurrences.

In this work, we propose to measure the quality of a concept based on its occurrence contexts. This is in fact a commonly used assumption in linguistics, known as "distributional hypothesis", which characterize the meaning of a word "by the company it keeps" [19]. Motivated by recent successes in embedding learning [20], [21], we propose to summarize the occurrence contexts information for each possible concept into an embedding vector representation, which we can then use to evaluate the quality of possible concepts, and better identify their occurrence in each local context. Our main contributions are as follows:

- We proposed an embedding based method for extracting a set of high quality concepts from corpus, by exhaustively generating concept candidates, learning their embedding vectors, and evaluating their qualities in the embedding space.
- We further recognize concept occurrences in texts, by comparing the embedding of concepts against the current local context, and select the one that is appropriate for the context.
- We experimentally demonstrate the effectiveness of our approach on three real-world datasets, covering the domains of machine learning, database and medicine.

II. Overview

We formalize the task of the concept mining as follows: Given a set of documents $\mathcal{D}$, each $d \in \mathcal{D}$ is a word sequence of the form $w_1 w_2 \ldots, w_n$, where concepts could correspond to individual word, or span across multiple words[1], the goal is to group words back together so that each grouped words correspond to the same concept[2] [3]. Specifically, there are two main outcomes we're interested in.

I) A *concept vocabulary* V_c that gathers together all concepts that ever occurred in corpus, each $c \in V_c$ is represented as a sequence of one or more words $w_1 w_2 \ldots w_{|c|}$[4]. To allow a more precise measure of concepts quality, a common practice is to associate each concept with a numeric measure of quality [2], [3], [17], [22]. We denote this as the concept quality scoring function $Q : V_c \rightarrow [0, 1]$. This task is referred to as *concept extraction*.

II) A *concept level representation* for each original document, as a consecutive sequence of concepts and words that occurred

[1]In this work, we do not consider the case where there are other words lying between the word sequence of a concept. It will be an interesting direction to explore in the future.

[2]This may come under different names, including phrase and single-word phrase [2], [3], or key-phrase [17], terminology [22], [14], multi-word expression [13] and simplex words, knowledge base entity [16], [23], [6]. The usage of term "concept" follows previous work [5], [24], [10].

[3]By *concept*, we mainly mean either single words or multiple word expression which have atomic meanings not derived from its components [13]. If restricted to noun type (noun or noun phrase), this is roughly equivalent to the notion of "concept" in [5], [24], [10], and the notion of "entity" in [16], [6], [23].

[4]Note that in this work we do not distinguish the string representation of a concept (i.e., a concept "mention" [16]) with the concept itself. In practice, we can preprocess the text using techniques such as lemmatization, word sense disambiguation [25], which is not the focus of this work.

in text, where each element in the sequence can be mapped to a subsequence in the original text, in a non-decreasing order. Specifically, for a word sequence $w_1 w_2 \dots w_n$, a concept level representation $c_1 \dots c_m$ is induced by two index sequences, $\mathbf{B} = b_1, \dots, b_m \leq n$, $\mathbf{E} = e_1, \dots, e_m$, so that $c_i = w_{b_i} \dots w_{e_i}$; If we use the shorthand $w_{[i,j]}$ to denote $w_i w_{i+1} \dots w_j$ for brevity, the concept level representation can be written as $w_{[b_1,e_1]} w_{[b_2,e_2]} \dots w_{[b_m,e_m]}$. This task is referred to as *concept recognition*.

Both of the above tasks rely on properly grouping words together in the original texts. The grouping process, however, needs to take into account not only the individual concept quality, but also their fitness to each other.

The grouping process can be described using the factor graph framework [26]. Given each original document $w_1 w_2 \dots w_n$, the probability takes the following form

$$P(\mathbf{B}, \mathbf{E} | w_1 w_2 \dots w_n) = \frac{1}{\Delta} \prod_{1 \leq i \leq m} f(w_{[b_i,e_i]}) \prod_{1 \leq i,j \leq m, i \neq j} g(w_{[b_i,e_i]}, w_{[b_j,e_j]}) \quad (1)$$

where the first type of feature function $f(w_{[b_i,e_i]})$ measures concepts' individual qualities, and the second type of feature function $f(w_{[b_i,e_i]}, w_{[b_j,e_j]})$ measures the fitness of each concept with respect to its neighbors; Δ is a normalization constant that makes the probabilities sum to one.

Solving this factor graph model is NP-hard [27]. Basically, there are an exponential number of choices of grouping words together that we need to examine in order to find the optimal solution.

To alleviate this, we will take a pragmatic approach. The intuition is that, we can first utilize existing techniques based on linguistic analysis, external knowledge bases, key-phrase extraction to generate a large pool of concept candidates; assuming concepts are captured by at least one of the approaches, we can then focus on candidate selection. By examining their occurrence contexts, and efficiently summarizing these information into embedding vectors, we're able to derive their global concept quality, and recognize their occurrences in the original text. The overall pipeline, as illustrated in Figure 1, consists of the following steps:

(i) We utilize existing techniques to generate a large number of concept candidates along with their occurrences in text. The result will be a large set of noisy, overlapping candidate occurrences.
(ii) We learn an embedding vector representation for each concept candidate based on its occurrence contexts and project all of them into an embedding space.
(iii) We learn a concept quality score function for candidates based on their embedding vectors. Each candidate will be associated with a score indicating how likely it is to be a concept.
(iv) We determine the true occurrence of concepts in each original document based on both their individual qualities and their fitness to context, according to the learned embedding vectors.

III. Candidate generation

We first generate candidates for further selection. The goal is to be exhaustive, so that recall is preserved and further selection process will not miss important candidates. While generating a large pool of candidate concepts from raw text is technically interesting, we mainly rely on existing techniques and do not make major contributions, and therefore will briefly discuss this step for completeness.

Specifically, as both the concept extraction and concept level representation depend on grouping word together in the original text, the generated candidates will take the form of *candidate concept occurrences*, where sequences of words denoting the same concepts are grouped together. We consider the following three sources of candidates:

- **Linguistic Analysis** We extract single words or word sequences as candidates if they matched the POS patterns defined in [4], or are detected by pre-trained noun phrase chunking model [28].
- **Knowledge Base**: We extract single words or word sequences if they are detected by the DBPedia Spotlight entity linking model [29].
- **State-of-the-arts**: We also incorporate the outputs from a set of state of art approaches, including statistical phrase mining [2], graph-based text summarization [17], supervised concept extraction [30] and key-phrase extraction [31]. Since some of the methods only generate a set of concepts without producing their occurrences in text, we generate occurrences for them via simple string matching.

Following previous approaches [2], [18], [16], [23], [32], We apply basic post-processing for the output of each above method, by keeping only noun phrases, i.e. those ending with a noun[5] and discard candidates with length above a threshold $K = 6$.

As illustrated in Figure 1, these candidate concepts occurrences will form a noisy, overlapping set, which can be viewed as a sequence of *super-concepts*. Each super-concept will be a word sequence that completely covers some of the candidates occurrences, and each candidate occurrence is covered by one and only one super-concept (The super-concepts are defined as the minimal ones that fit the above requirement). Given a word sequence $w_1 w_2 \dots w_n$, we denote a super-concept sequence $\mathbf{U} = u_1 u_2 \dots u_l$ using the following two index sequences, $\mathbf{B}' = b'_1, \dots, b'_l$, $\mathbf{E}' = e'_1, \dots, e'_l$, so that each $u_i = w_{[b'_i,e'_i]}$. As an example, in the case where the candidate occurrences "vector machine" and "support vector machine" overlap, then "support vector machine" would be a super-concept that contains both candidates. And the task is to make a choice between the two.

[5]Extracting other types of phrases, e.g. verb phrases, adjective phrases may also be interesting. However, our approach is not specific to noun phrases and we can simply replace the above noun phrases filters to detect different type of phrases.

In other words, we have simplified our task (see Equation 1) as selecting candidates from each *super-concept*, instead of exhaustively searching all different ways of grouping words together. If we use variables $\mathbf{Z} = z_1 z_2 \ldots, z_l$ to denote the candidate selection decision in each super-concepts, where z_i denotes the unit (a candidate or a plain text word) in u_i we choose to be included in the concept level representation, then Equation 1 can be simplified as

$$P(u_1 u_2 \ldots u_l, \mathbf{Z}) = \frac{1}{\Delta} \prod_{1 \le i \le l} f(z_i) \prod_{1 \le i,j \le l, i \ne j} g(z_i, z_j) \mathbb{I}(z_i \in u_i) \tag{2}$$

$\mathbb{I}(\cdot)$ is the identity function that returns 1 if the condition is true, and 0 otherwise.

The selection of candidates, however, needs to consider all the candidates at a *global level*: How can we best leverage the global occurrence contexts to judge the concept qualities? And how can we use these information to best recognize concepts' occurrences in local contexts? These will be the key questions to address in the next section.

IV. Embedding aware concept mining

In this section, we describe our approach to produce the concept vocabulary and concept level representations based on the generated candidates. We will start by describing the objective function and introducing our overall approach (subsection IV-A), and then discuss each individual step in detail (subsection IV-B - subsection IV-D).

A. Framework

The goal of our concept mining approach is to maximize both the individual qualities of the selected concepts and their fitness to the surrounding contexts. To that end we propose to use the following feature functions (see Equation (1), (2)) to quantify our objective.

First, we study the feature function for each node pair, $g(z_i, z_j)$, which accounts for the context fitness. We will start by introducing a new set of parameters θ, each θ_z, as indexed by either a word or a concept candidate z, is a vector of fixed length (For convenience we refer to the set of all words and concept candidates as $\mathcal{V}$). There are several commonly used models to measure embedding based context fitness, e.g. Softmax, Hierarchical Softmax, Skip-Gram with Negative Sampling (SGNS) (see [20]), Adaptive Softmax (see [33]). In their simplest form, Softmax, our fitness measure $g(z_i, z_j)$ can be expressed as

$$g(\mathbf{z_i}, \mathbf{z_j}|\theta) = \begin{cases} \frac{\exp(\theta_{\mathbf{z_i}} \cdot \theta_{\mathbf{z_j}})}{(\sum_{\mathbf{z'} \in \mathcal{V}} \exp(\theta_{\mathbf{z_i}} \cdot \theta_{\mathbf{z'}}))^{\frac{1}{2}} (\sum_{\mathbf{z'} \in \mathcal{V}} \exp(\theta_{\mathbf{z_j}} \cdot \theta_{\mathbf{z'}}))^{\frac{1}{2}}} & |i-j| \le K \\ 1 & |i-j| > K \end{cases} \tag{3}$$

The above definition of the node feature function is in accordance with the word2vec model [20]: for each pair of units from the super-vocabulary (e.g. words or concept candidates) that is within a local window of size K, where K is a predefined constant, we measure the proximity of their vector representation based on the Softmax function. We can see that the embedding is indeed learned from the context: the embedding of a word or concept candidate is completely determined by the contexts it occurs in, regardless of what it originally is.

As for the context-independent node feature function $f(z_i)$, we will directly define it as the concept quality score[6].

$$f(z_i) = Q(z_i) \tag{4}$$

We can now formally describe our objective function according to Equation 2 and the feature function defined above, as a function of the embedding vectors θ, concept quality score function Q, and the selection decisions in the super-concepts sequence $\mathbf{Z}^{(d)} = z_1^{(d)} z_2^{(d)} \ldots, z_{l^{(d)}}^{(d)}$ for each document $d \in \mathcal{D}$, shorthanded as $\mathcal{Z} \triangleq \{\mathbf{Z^{(d)}} | d \in \mathcal{D}\}$. Specifically, it takes the following form

$$O(\theta, \mathcal{Z}, Q) = \prod_{d \in \mathcal{D}} \Big(\prod_{1 \le i \le l^{(d)}} Q(z_i^{(d)}) \cdot \prod_{1 \le i,j \le l^{(d)}, i \ne j} g(z_i^{(d)}, z_j^{(d)} | \theta) \cdot \prod_{1 \le i \le l^{(d)}} \mathbb{I}(z_i^{(d)} \in u_i^{(d)}) \Big) \tag{5}$$

For learning the embeddings θ, the concept quality scores $Q(\mathbf{z_i})$ and the recognized candidate occurrences $\mathcal{Z}$, we take a 3-step approach: we firstly learn the embedding parameters θ by maximizing the *expectation* of the objective function (subsection IV-B); then we learn concept quality score function Q (subsection IV-C) incorporating the information from the learned embeddings θ; and finally, we treat the learned embedding θ and concept quality score function Q as fixed, and perform an exact optimization of the objective function $O(\theta, \mathcal{Z}, Q)$ to obtain selection choices $\mathcal{Z}$. The rationale is the following: since we have no prior knowledge of which candidate to choose for each of the overlapping region, we will assume it is uniformly distributed, and learns the embedding of each concept candidates that equally reflect all of its possible occurrence contexts; likewise, the quality score of each concept, should be an averaged ensemble where each possible occurrence context contributed equally; and finally, we use these averaged ensembles to determine context fitness and select the best settings.

B. Concept Embedding

We first learn the embedding θ, by maximizing towards the *expectation* of the objective function. Assuming the concept selection choices $\mathcal{Z}$ is randomly distributed, where each of its component value, $z_i^{(d)}$ for $d \in \mathcal{D}$ is randomly select

[6] The concept quality score is defined as a small enough positive constant for those inputs that are not concepts. The exact value does not matter to the optimization process.

from available choices for the specific super-concept, the maximization goal becomes

$$\max_{\theta} \mathop{\mathbb{E}}_{\mathcal{Z},Q}[O(\theta,\mathcal{Z},Q)] \tag{6}$$
$$=\frac{1}{\Delta}\sum_{\mathcal{Z}}(\prod_{d\in\mathcal{D}}(\prod_{1\le i,j\le l^{(d)},i\neq j} g(z_i^{(d)},z_j^{(d)}|\theta) \cdot \prod_{1\le i\le l^{(d)}} \mathbb{I}(z_i^{(d)}\in u_i^{(d)})))$$

Again, Δ is a constant accounting for taking the expectation over Q and $\mathcal{Z}$, and is irrelevant for the optimization process.

Similar to vanilla word embedding learning, we learn the embedding vectors in a stochastic fashion [20], where at each time we maximize the feature function g with respect to a specific pair $(z_i^{(d)}, z_j^{(d)})$, and keep iterating over all such possible pairs.

C. Concept Quality Estimation

In this step, each concept candidate is associated with an embedding vector, and thus mapped to a point in the embedding space. Our goal is to measure their quality in the embedding space, and produce the concept quality scoring function $Q(c) : V_c \to [0,1]$[7]. We will propose several novel measures to capture the quality of a concept's occurrence contexts via its embedding. The idea is originated from our usage experience with previous phrase mining tools, such as Autophrase: it is able to produce high quality concepts such as "query plan", whose learned embedding is similar to many other high quality concepts such as "join operators", "join orders", and "query optimizer"; but it may also generate low quality ones such as "following treatment", which do not have that many high quality neighbors.

To that end, we examine the two most used properties in the intrinsic evaluation of semantic embedding vectors [20], [34]: semantic similarity, e.g. answering questions such as "what is the most similar word to king?"; and semantic relatedness, e.g. answering questions such as "men is to women as king is to what?". It has been shown that relatedness can be derived from similarity using vector arithmetic [34]. Therefore, we will solely focus on semantic similarity. Specifically, we measure the similarity of two embedding vectors θ_c, $\theta_{c'}$, denoted as $\langle\theta_c,\theta_{c'}\rangle$, using the cosine similarity function

$$\langle\theta_c,\theta_{c'}\rangle = \frac{\theta_c\cdot\theta_{c'}}{||\theta_c||||\theta_{c'}||} \tag{7}$$

where $\cdot$ denotes vector inner product and $||v||$ denotes the l2-norm of vector. We advocate a spatial representation of the embedding vectors, and focus on what we call a concept's "similarity neighborhood", that is, all concepts that have similarity above a certain threshold κ with the concept of interest[8]. We propose the following four types of quality measure.

Context commonness measures how common a concept's occurrence contexts are with respect to other concepts. For example, the concept "support vector machine" may often be associated with *common* context patterns that are also suitable for other machine learning algorithms, such as "random forest" or "logistic regression". On the other hand, for a less meaningful candidate concept, e.g. "vector support machine", it will probably only appear in some rarely seen contexts, which can not be associated with that many concepts.

We define the context commonness as the number of members in a concept c's neighborhood : the higher this value is, the more concepts there will be that are associated with its similar contexts, and hence the more common its contexts should be

$$\sum_{c'\in V_c, c'\neq c} \mathbb{I}(\langle\theta_c,\theta_{c'}\rangle > \kappa) \tag{8}$$

Context purity focuses on the internal differences between a concept's occurrence contexts. For example, for a concept with clear meanings such as "query optimizer", its usage contexts are usually quite specific and often associated with the same type of concepts; while casual expressions such as "nice property" will have a much more diverse set of usage contexts, and thus less "pure".

We define context purity as the average similarity between a concept and concepts in its similarity neighborhood. High value means that its usage contexts are more likely to be associated with only specific type of concepts

$$\frac{\sum_{c'\in V_c, c'\neq c}\langle\theta_c,\theta_{c'}\rangle}{\sum_{c'\in V_c, c'\neq c}\mathbb{I}(\langle\theta_c,\theta_{c'}\rangle > \kappa)} \tag{9}$$

Context link-ability measures to what extent a concept's occurrence context can be associated with existing high quality concepts in the knowledge base. For example, if we know that "rhinella marina" (a type of toad) usually occurs in contexts that are similar to"frog", and that "frog" already exists in the knowledge base as a high quality concept, it would help us confirm that "rhinella marina" is also a high quality concept.

We define context link-ability as the number of concepts in a concept c's neighborhood belonging to a predefined set of high quality concepts V_{ext}

$$\sum_{c'\in V_{ext}, c'\neq c} \mathbb{I}(\langle\theta_c,\theta_{c'}\rangle > \kappa) \tag{10}$$

A nice property of this measure is that, the knowledge base entities themselves need not have high value in this measure: they may or may not be similar to *other* entities in the knowledge base. This will allow us to re-use them as training examples for the concept quality score model.

Context generalizability measures whether the string representation of a concept can be generalized to denote more

[7]In this work we simply take the set of all concept candidates as the set of concepts V_c, and rely on the concept quality scoring function to compute distinguishes the high quality concepts from low quality ones. As a result, we may use the term "concept" and "candidate" interchangeably.

[8]κ can be determined from dataset, e.g., by examining the lists of most similar concepts for each concept (or some of the concepts), and determining a cutoff threshold

specific ones. As an example, if we see two or more concepts (but not one) such as "fuzzy support vector machine", "twin support vector machine", "one-class support vector machine", all of which have similar usage context as "support vector machine", it would further confirm that "support vector machine" is a valid concept, rather than a random word combination.

We define context generalizability as the number of concepts in a concept c's neighborhood that contains it as subsequence:

$$\max_{\theta}(\sum_{c' \in V_c, c \prec c'} \mathbb{I}(\langle \theta_c, \theta_{c'} \rangle > \kappa) - 1, 0) \tag{11}$$

where we use $c \prec c'$ to denote that c is a subsequence of c'.

Based on the measures defined above, we produce a combined score using a machine learning model that adaptively weight each features, similar to previous approaches [3], [2]. Specifically, given a few positive examples, e.g. concepts that occur in knowledge bases, and optionally a few negative examples [2], we will train a classification model such as random forest, or support vector machine, and use its predictions as the quality scores.

D. Concept Recognition

Finally, in the concept recognition step, we compute $\mathbf{Z}$ and determine which concept to recognize for each super-concept. Specifically, the task is to maximize Equation 5 with respect to $\mathbf{Z}$. The challenge lies in the fact that the decisions of concept occurrence may influence each other, and a brute-force approach to find the exact maximum solution may take exponential time. In the following, we present our efficient inference approach that is guaranteed to reach the global optimal result in linear time.

To achieve this, we explore the locality of the correlation in the overall objective function, and write maximization goal as

$$\begin{aligned} &\max_{\mathcal{Z}} O(\theta, \mathcal{Z}, Q) \\ =&\frac{1}{\Delta} \prod_{d \in \mathcal{D}} (\prod_{1 \le i \le l^{(d)}} Q(z_i^{(d)}) \cdot \prod_{1 \le i,j \le l^{(d)}, i \ne j} g(z_i^{(d)}, z_j^{(d)} | \theta) \\ &\cdot \mathbb{I}(z_i^{(d)} \in u_i^{(d)})) \end{aligned} \tag{12}$$

It can be easily seen that the above can be decomposed into the maximization within each document. Specifically, given each one document $u_1 u_2 \dots u_l$, the goal is to output the best configuration for that document, $\mathbf{Z} = z_1 z_2 \dots z_l$.

We apply a technique called junction tree algorithm [26]. The basic idea is to re-parameterize the parameters to make the maximization easy. We will introduce a variable to represent each local window $z_{t-K+1} z_{t-K+2} \dots z_t$ of size K, for $t = K, K+1, \dots l$. Without confusion we denote it as the *variable* $\mathbf{Z}_{t:-K}$. It can be shown that the new factor graph can be described by the following factors:

$$h_{t,t+1}(\mathbf{Z}_{t:-K}, \mathbf{Z}_{t+1:-K}) = f(z_{t+1}) \prod_{i=t-K+1,\dots,t} g(z_i, z_{t+1}) \tag{13}$$

for $t = K, K+1, \dots l-1$, and an additional factor accounting for the initial condition

$$h_0(\mathbf{Z}_{K:-K}) = \prod_{i=1,\dots,K} f(z_i) \prod_{i=0,\dots,K-1} \prod_{j=i+1,\dots,K} g(z_i, z_j) \tag{14}$$

Now that the new factor graph forms a linear chain structure, we can easily obtain the optimal setting. Specifically, we apply the max-product inference procedure [26], and the message passing rule and factors between nodes can be expressed as

$$\mu_{\mathbf{Z}_{t:-K} \to h_{t,t+1}}(\mathbf{Z}_{t:-K}) = \sum_{f \in ne(\mathbf{Z}_{t:-K}) \setminus h_{t,t+1}} \mu_{f \to \mathbf{Z}_{t:-K}}(\mathbf{Z}_{t:-K}) \tag{15}$$

where $ne(\mathbf{Z}_{t:-K})$ refers to all "neighboring" factors associated with variable $\mathbf{Z}_{t:-K}$, and

$$\begin{aligned} \mu_{h_{t,t+1} \to \mathbf{Z}_{t+1:-K}}(\mathbf{Z}_{t+1:-K}) = &\max_{\mathbf{Z}_{t:-K}} \\ &(h_{t,t+1}(\mathbf{Z}_{t:-K}, \mathbf{Z}_{t+1:-K}) \cdot \mu_{\mathbf{Z}_{t:-K} \to h_{t,t+1}}(\mathbf{Z}_{t:-K}) \\ &\cdot \mathbb{I}((\mathbf{Z}_{t+1:-K})_{1:K-1} = (\mathbf{Z}_{t:-K})_{2:K}) \quad) \end{aligned} \tag{16}$$

where the last term is to ensure that the re-parameterized variables $\mathbf{Z}_{t+1:-K}$ and $\mathbf{Z}_{t:-K}$ agree on the overlapping region, so that the original variables can be recovered. And the initial condition is given by

$$\mu_{h_0 \to \mathbf{Z}_{K:-K}}(\mathbf{Z}_{K:-K}) = h_0(\mathbf{Z}_{K:-K}) \tag{17}$$

We present the inference algorithm in Algorithm 1.

Algorithm 1 Concept Recognition

Input: super-concept sequence $U = u_1 u_2 \dots u_l$, factors $f : \mathcal{V} \to \mathbb{R}$, $g : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$, window size K
Output: : word sequences to identify within each super-concept sequence $\mathbf{Z}_{1:l} = z_1 z_2 \dots z_l$
initialize $\mu_{h_0 \to \mathbf{Z}_{K:-K}}(\mathbf{Z}_{K:-K})$ according to Equation 17
for $t = K$ **to** $l - 1$ **do**
  update $\mu_{\mathbf{Z}_{t:-K} \to h_{t,t+1}}(\mathbf{Z}_{t:-K})$ according to Equation 15
  update $\mu_{h_{t,t+1} \to \mathbf{Z}_{t+1:-K}}(\mathbf{Z}_{t+1:-K})$ according to Equation 16
end for
compute $(\mathbf{Z}_{n:-K})_{max}$ to maximize the last messages, as $\arg\max_{\mathbf{Z}_{t:-K}} \mu_{h_{n-1,n} \to \mathbf{Z}_{t:-K}}(\mathbf{Z}_{t:-K})$
for $t = l - 1$ **to** K **do**
  find $(\mathbf{Z}_{t:-K})_{max}$ that maximizes the right hand side of Equation 16, given $\mathbf{Z}_{t+1:-K} = (\mathbf{Z}_{t+1:-K})_{max}$
end for
return $\mathbf{Z}_{1:l}$

As shown in [26], the junction tree inference algorithm is able to achieve the optimal results; it is also quite efficient, since it only uses $\mathcal{O}(l)$ iterations to find the optimal value as well as the corresponding parameter setting, where l is the number of latent variables $z_1 z_2 \dots z_l$.

TABLE II: Dataset Statistics

	Machine Learning	Database	Medicine
# docs	14.5K	3.6K	110K
# sentences	5.9M	2.2M	0.9M
# words	77M	31M	15M

V. Experiment

In this section, we experimentally demonstrate the effectiveness of our proposed approach in 3 benchmark dataset, from the domains of machine learning, database and medicine.

A. Experiment Setup

1) Datasets: We have collected the following 3 datasets. Their statistics are summarized in Table II.

Machine Learning The first dataset contains a complete full-text collection till 2017 of the NIPS proceeding[9], JMLR journal[10] and the Proceedings of Machine Learning Research (PMLR), which includes proceedings from major machine learning conferences such as ICML, AISTATS and COLT.

Database Our second dataset contains complete full-text collection till 2017 of VLDB[11] and SIGMOD proceeding[12], the two top publication venue in the field of database.

Medicine We have also collected a paper abstract dataset from PubMed[13] in the domain of medicine. Specifically, the 31 medicine journal that has more than 1M bytes of words and SCI impact factor above 1, which includes prominent ones such as The New England journal of medicine, JAMA, British medical journal and Lancet.

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

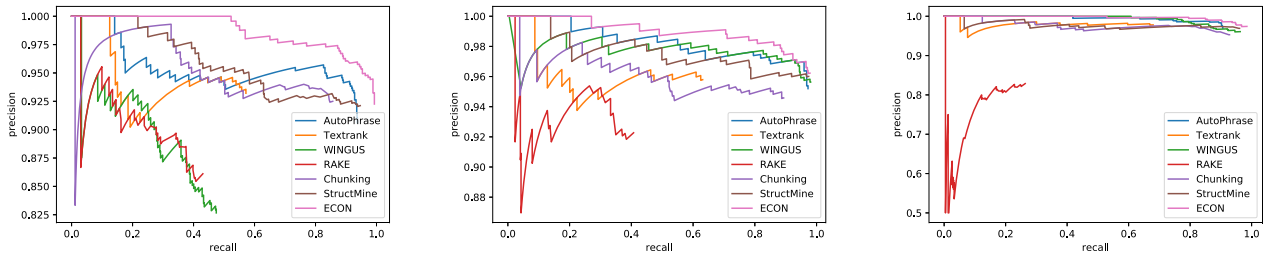
- **AutoPhrase** [2] is the state-of-the-art phrase mining technique, which combines the quality estimation and occurrence identification to extract salient phrases from text documents with little to zero amount of human labeling. We follow the settings recommended by the original paper [2] and the released code[14].
- **Textrank** [17] is an unsupervised key-phrase extraction algorithm that leverages the graph based importance to select significant key-phrases from documents. We use the implementation and the parameter settings according to a publicly available version[15], and keep the top 20% of the extracted key-phrases. Since the entire corpus is too long to fit in memory, we batchify the input data, 20 sentences at a time, and sum the scores of all the documents for each keyphrase as their final score.
- **WINGUS** [35] is a supervised key-phrase extraction approach which considers a large set of grammatical, statistical and document logical structure features. We use the implementation and the parameter settings according to a publicly available version[16] and batchify the corpus similar to the above.
- **Chunking** directly captures the occurrences of the word sequences of interest as a textual span. There are two choices, the noun phrase chunking model and named entity recognition model. Because we observe that noun phrase chunking gives better performance, and also because it's more related to our task, we report the performance of the noun phrase chunking model, implemented in Spacy [28], as the performance of the chunking approach.
- **Grammartical Pattern Extraction** captures concepts using pre-defined regular expression over POS tags of the word sequence. We used the grammar pattern according to the publicly available software, associated with [4], "StructMine"[17]. We refer to this approach using the name "StructMine".
- **RAKE** [31] employs the phrase delimiter to detect candidate phrases, estimates their quality using a graph-based importance measure, and then forms more phrases based on their adjacency in text. We used the implementation and the parameter settings in a publicly available version[18].
- **ECON** refers to our approach, Embedding based CONcept mining. For the objective function, the window size is set to 5, and the node pair feature function is implemented using the Hierarchical Softmax model [20]. For concept quality estimation step, we use a linear support vector machine implementation from Scikit-learn[19], with positive training example provided by Dbpedia and equal number of randomly sampled negative training samples to produce the quality score function;

For methods that generate textual occurrences but do not provide quality scores, we use the sum of TF-IDF values across all the documents as the concept quality scores. For methods that only output a plain set of concepts, but not textual occurrence, we generate occurrence of these concepts via simple string match.

In order to have a fair comparison, we did some basic filtering on the set of extracted concepts, to discard all extracted concepts not ending with a noun, or starting with a determiner other than "the", which helps to improve the accuracy of baseline methods.

3) Evaluation Metrics: We measure the quality of the extracted concepts from each method in terms of precision and recall. Because we observed that common extracted phrases that rank high in each method can be easily captured by many other state of the art phrase mining and keyword extraction methods, we focus on *long tail* concepts. Specifically, for the concepts generated by each of the methods, we exclude the ones that can also be found by others (which are easier to find), and then, randomly sample $k = 100$ elements from the rest of the extracted concepts for each method, and merge them into a ground-truth evaluation set. We then manually evaluate the

[9]https://nips.cc/
[10]http://www.jmlr.org/
[11]http://www.vldb.org/
[12]https://sigmod.org/conferences/
[13]https://www.ncbi.nlm.nih.gov/pubmed/
[14]https://github.com/shangjingbo1226/AutoPhrase
[15]https://github.com/summanlp/textrank
[16]https://github.com/boudinfl/pke
[17]https://github.com/shanzhenren/StructMineDataPipeline
[18]https://github.com/fabianvf/python-rake
[19]http://scikit-learn.org/



(a) Machine Learning (b) Database (c) Medicine

Fig. 2: Precision-recall curves for concept quality evaluation

TABLE III: Example concept level representation generated by various approaches

ECON	AutoPhrase	StructMine
Fast **[global convergence]** of **[gradient methods]** for high-dimensional **[statistical recovery]**	**[Fast global convergence of gradient methods]** for high-dimensional statistical recovery	**[Fast global convergence]** of **[gradient methods]** for **[high-dimensional statistical recovery]**
Our **[paper]** develops **[Riemannian LBFGS]**, which can also achieve local **[superlinear convergence]**.	Our **[paper develops]** Riemannian **[LBFGS]**, which can also achieve local **[superlinear convergence]**.	Our **[paper]** develops **[Riemannian LBFGS]**, for which can also achieve **[local superlinear convergence]**.
To incorporate these **[results]** into the corresponding **[parent workflow]**, the expected **[turnaround time]** of the **[parent]** is calculated in the following **[hierarchical manner]**.	To incorporate these results into the corresponding parent workflow, the expected turnaround time of the parent is calculated in the following **[hierarchical manner]**.	To incorporate these **[results]** into the **[corresponding parent workflow]**, the expected **[turnaround time]** of the **[parent]** is calculated in the **[following hierarchical manner]**.
If current **[mortality trends]** continue, **[HIV AIDS]** can be expected to become one of the five **[leading causes]** of **[death]** by 1991 in **[women]** of **[reproductive age]**	If current **[mortality trends]** continue, **[HIV AIDS]** can be expected to become one of the five leading causes of death by 1991 in women of reproductive age .	If **[current mortality trends]** continue, **[HIV AIDS]** can be expected to become one of the five **[leading causes]** of **[death]** by 1991 in **[women]** of **[reproductive age]**.

quality of ground-truth evaluation set, based on a principle we call "knowledge-baseness": if we were to construct a Wiki-like knowledge base from the target domain, is it significant enough, that it's worthwhile to be explained and added to the knowledge base? Google and domain dictionary could help, but for niche ones like "APCA" (a time series analysis technique) that carries specific meanings, we also want to include them as true concepts.

In addition, we evaluate the quality of concept identification using the application task of information retrieval. Specifically, given a concept as query, we aim to retrieve relevant sentences based on the result of concept identification. Similar to the above, we take the top $K = 50$ retrieved documents from each method, exclude common ones that occurred in all of them, and merge them to form the ground truth documents. We then manually label them using the same standard as above and measure their performance using the Mean Average Precision (MAP)[20] score, averaged over 5 concept queries. The queries are randomly selected from commonly known concept, which can usually be extended with prefix and suffix, e.g. "belief propagation" as in "tensor belief propagation", "global convergence" as in "global convergence rate".

B. Results

Precision and recall evaluation Figure 2 presents the precision-recall curves of all compared methods. The performances vary by dataset, with the medicine domain being the easiest, where the terminologies are rigorous and the extracted concepts for each method are quite meaningful in general; the machine learning domain is the most noisiest one, where longer and more complex word sequences such as "maximal ancestral graph" are required to express a concept. In general, we observe that different approaches define the quality measure in their own way, which appears to be consistent across datasets: StructMine and Chunking are the most straightforward, that counts the number of times a word sequence occur, "standalone" (as opposed to being contained in a longer sequence); Textrank emphasizes on the connection to neighbors, and can capture popular concepts such as "vertex"; the supervised method, WINGUS, performs relatively well, however it sometimes misses important ones such as "NMSE" and gives false positive results such as "such transformations"; RAKE only captures multi-word phrases and completely misses single-word ones, and tends to produce long phrases such as "respiratory tract cultures yielding aspergillus" (medicine); Autophrase is good at extracting coherent phrases, such as "minimum inhibitory concentration" (medicine), but is not good at capturing atomic single-word concepts such as "matrix" (machine learning). ECON ranks concepts in a more natural way, from obvious ones such as "convolution", "covariate matrix" down to borderline cases such as "unsupervised grammar induction", and then down to low quality ones such as "feedback setting". However, in the medicine dataset, it tends to give high scores to expressions like "essential role" and "unmet need" which always appear in consistent contexts. This, along with the fact that medical terms are easy to capture

[20]https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

TABLE IV: MAP scores for information retrieval evaluation

Method	Machine Learning	Database	Medicine
AutoPhrase	0.87	0.87	0.87
Textrank	0.84	0.92	0.81
WINGUS	0.89	0.92	0.85
RAKE	0.77	0.67	0.58
StructMine	0.91	0.93	0.85
Tagger	0.90	0.82	0.82
ECON	**0.98**	**0.99**	**0.97**

TABLE V: Example concepts captured by ECON

Dataset	Example Concepts
Machine Learning	variable, eigenvalue, likelihood, matrix, value injection experiment, move-making algorithm
Database	locks, XES, overhead, throughput, preference relation, System Administrator, distributed file system
Pubmed	antigens, HBP, titre, prostate, TUDCA, proportional hazards regression, fragment length polymorphism

using co-occurrence and simple grammatical rules, causes its performance to drop and be very close to the best performing baseline methods in that dataset.

Information retrieval task evaluation The MAP score for each method is shown in Figure 2. ECON achieves high recognition accuracy in general; the main drawback is that it tends to favor atomic ones, recognizing "bone marrow transplantation" in place of "autologous bone marrow transplantation". Other methods tend to fluctuate between each query, since they mostly ignore context words and grammatical constraint; for those that are based on grammatical patterns, they have difficulty distinguishing between bad ones such as "a leading commercial centralized relational database", and good ones such as "generalized belief propagation".

C. Qualitative Evaluation

We present the following qualitative evaluation to further compare the proposed approach against baseline methods.

Concepts Representation Table III illustrates the concept level representation generated by ECON, Autophrase and the pattern based approach, StructMine, without noun phrases filtering on the recognition results. We can observe that, ECON can capture concept occurrences in a more comprehensive fashion, adaptively including or excluding decorative words depending on the contexts; Autophrase only selectively labels a few concept occurrences, which may or may not conform to grammar patterns; StructMine, on the other hand, generates concept representation by greedily including all decorative words for each noun phrases.

Concept quality difference We compare the results of concept quality score produced by our approach and the statistical signal based phrase mining approach, Autophrase. Table V shows some example concepts that can be confidently captured by our approach but not by Autophrase. Specifically, we compare the rank of concepts among all evaluated concepts between our approach and Autophrase, and select those that have a difference bigger than 50 %. We can observe that, our approach can better capture general, but meaningful concepts such as "likelihood"; it can also better capture concepts made up of frequent words, which may be missed by statistical co-occurrence based measures.

VI. Related Work

Historically, the problem of mining concept from texts is addressed within the NLP community [23], [13], [36], [37], [38], with the closest lines of work being noun phrase chunking and named entity recognition [23], which either employ heuristics such as fixed POS-tag patterns [4], or use large amount of labeled training data and train complex models based on CRF, LSTM [15] and CNN [39].

Specifically, there has been a line of work focusing on terminology and keyphrase extraction, where they generated noun phrases as candidates, and exploited statistical occurrence and co-occurrence measures [22], semantic information from knowledge base [40], along with other textual features such as whether the candidate appears in title or abstract, whether it is an acronym, or whether it ends with specific suffix in order to rank the candidates [30]. Supervised methods aim at replacing human effort in assigning keywords to each document using the above features, while unsupervised ones typically use similarity graph between candidates in order to compute importance score and generate the top concepts for each document [18], [31], [17]. This is quite different from our goal.

The state of art approaches for mining quality phrases is [3], [2]. They innovatively proposed the approach of adaptively recognizing concept occurrence based on concept quality, and exploited various important statistical features such as popularity, concordance, informativeness [3], as well as POS-tag sequence [2] to measure phrase quality, and leverage knowledge base entity names as training label, to train the concept quality scoring function. They mainly focus statistically significant phrases, not concepts with low occurrences and co-occurrences.

VII. Conclusion

In this work, we studied the problem of concept mining, where given a set of documents, the goal is to extract concepts and recognize their occurrences. We proposed a novel framework that first exhaustively generate candidates, and then perform candidate selection based on embeddings. Many interesting future directions exists. One is to further explore the relations between these mined concepts to extract more structured information from text. Another direction is to incorporate more supervised signal, such as performance in downstream tasks to enhance the concept mining process.

VIII. Acknowledgement

This research was sponsored in part by a VISA research grant and the Army Research Laboratory under cooperative

agreements W911NF09-2-0053. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

REFERENCES

[1] M. Ware and M. Mabe, "The STM report: An overview of scientific and scholarly journal publishing," 2015.

[2] 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*, 2018.

[3] J. Liu, J. Shang, C. Wang, X. Ren, and J. Han, "Mining quality phrases from massive text corpora," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1729–1744.

[4] X. Ren, Z. Wu, W. He, M. Qu, C. R. Voss, H. Ji, T. F. Abdelzaher, and J. Han, "Cotype: Joint extraction of typed entities and relations with knowledge bases," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 1015–1024.

[5] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: A probabilistic taxonomy for text understanding," in *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, 2012, pp. 481–492.

[6] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *Proceedings of the 20th international joint conference on Artifical intelligence*, 2007, pp. 2670–2676.

[7] M. Miwa and M. Bansal, "End-to-end relation extraction using lstms on sequences and tree structures," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2016, pp. 1105–1116.

[8] 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*, 2018, pp. 37–45.

[9] 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*, 2017, pp. 1271–1279.

[10] N. Dalvi, R. Kumar, B. Pang, R. Ramakrishnan, A. Tomkins, P. Bohannon, S. Keerthi, and S. Merugu, "A web of concepts," in *Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, 2009, pp. 1–12.

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

[12] R. Jackendoff, *The architecture of the language faculty*. MIT Press, 1997, no. 28.

[13] T. Baldwin and S. N. Kim, "Multiword expressions." *Handbook of natural language processing*, vol. 2, pp. 267–292, 2010.

[14] J. S. Justeson and S. M. Katz, "Technical terminology: some linguistic properties and an algorithm for identification in text," *Natural language engineering*, vol. 1, no. 1, pp. 9–27, 1995.

[15] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[16] X. Ren, A. El-Kishky, H. Ji, and J. Han, "Automatic entity recognition and typing in massive text data," in *Proceedings of the 2016 International Conference on Management of Data*, 2016, pp. 2235–2239.

[17] R. Mihalcea and P. Tarau, "Textrank: Bringing order into text," in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[18] S. Beliga, A. Meštrović, and S. Martinčić-Ipšić, "An overview of graph-based keyword extraction methods and approaches," *Journal of information and organizational sciences*, vol. 39, no. 1, pp. 1–20, 2015.

[19] J. R. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in linguistic analysis*, 1957.

[20] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2013.

[21] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2014.

[22] K. Frantzi, S. Ananiadou, and H. Mima, "Automatic recognition of multi-word terms:. the c-value/nc-value method," *International Journal on Digital Libraries*, vol. 3, no. 2, pp. 115–130, 2000.

[23] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[24] A. Parameswaran, H. Garcia-Molina, and A. Rajaraman, "Towards the web of concepts: Extracting concepts from large datasets," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 566–577, 2010.

[25] R. Navigli, "Word sense disambiguation: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 2, p. 10, 2009.

[26] M. J. Wainwright, M. I. Jordan *et al.*, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[27] M. R. Garey and D. S. Johnson, "Computers and intractability: A guide to the theory of npcompleteness (series of books in the mathematical sciences), ed," *Computers and Intractability*, p. 340, 1979.

[28] M. Honnibal and I. Montani, "spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing," *To appear*, 2017.

[29] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "Dbpedia spotlight: shedding light on the web of documents," in *Proceedings of the 7th international conference on semantic systems*, 2011, pp. 1–8.

[30] T. Nguyen and M.-Y. Kan, "Keyphrase extraction in scientific publications," *Asian Digital Libraries. Looking Back 10 Years and Forging New Frontiers*, pp. 317–326, 2007.

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

[32] Z. Zhang, J. Iria, C. Brewster, and F. Ciravegna, "A comparative evaluation of term recognition algorithms," 2008.

[33] E. Grave, A. Joulin, M. Cissé, D. Grangier, and H. Jégou, "Efficient softmax approximation for gpus," *arXiv preprint arXiv:1609.04309*, 2016.

[34] O. Levy and Y. Goldberg, "Linguistic regularities in sparse and explicit word representations," in *Proceedings of the eighteenth conference on computational natural language learning*, 2014, pp. 171–180.

[35] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: Practical automatic keyphrase extraction," in *Proceedings of the fourth ACM conference on Digital libraries*, 1999, pp. 254–255.

[36] P. Pecina and P. Schlesinger, "Combining association measures for collocation extraction," in *Proceedings of the COLING/ACL on Main conference poster sessions*, 2006, pp. 651–658.

[37] C. Sporleder and L. Li, "Unsupervised recognition of literal and non-literal use of idiomatic expressions," in *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 2009, pp. 754–762.

[38] C. Hashimoto and D. Kawahara, "Construction of an idiom corpus and its application to idiom identification based on wsd incorporating idiom-specific features," in *Proceedings of the conference on empirical methods in natural language processing*, 2008, pp. 992–1001.

[39] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[40] O. Medelyan and I. H. Witten, "Thesaurus based automatic keyphrase indexing," in *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, 2006, pp. 296–297.