CONCEPT AWARE CO-OCCURRENCE AND ITS APPLICATIONS

by

Klemen Simonic

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computing

School of Computing

The University of Utah

August 2015

The University of Utah Graduate School

STATEMENT OF THESIS APPROVAL

The thesis of **Klemen Simonic**

has been approved by the following supervisory committee members:

Feifei Li	, Chair	**5/6/2015** Date Approved
Jeffrey Phillips	, Member	**5/6/2015** Date Approved
Vivek Srikumar	, Member	**5/6/2015** Date Approved

and by **Ross Whitaker** , Chair/Dean of

the Department/College/School of **Computing**

and by David B. Kieda, Dean of The Graduate School.

ABSTRACT

Term co-occurrence data has been extensively used in many applications ranging from information retrieval to word sense disambiguation. There are two major limitations of co-occurrence data. The first limitation is known as the data sparseness problem or the zero frequency problem: For a majority of pairs, the probability that they co-occur in even a large corpus is very small. The second limitation is that in co-occurrence data, each term is considered as a meaningless symbol, or in other words, terms do not have types, or any semantic relationships with other terms. In this paper, we introduce a novel approach to address these two limitations. We create concept aware co-occurrence data wherein each term is not a symbol, but an entry in a large-scale, data-driven semantic network. We show that with concepts or types, we are able to address the data sparseness problem through generalization. Furthermore, using concept co-occurrence, we show that our approach can benefit a large range of applications, including short text understanding.

CONTENTS

ABSTRACT . . . . . **iii**

ACKNOWLEDGMENTS . . . . . **vi**

CHAPTERS

1. INTRODUCTION **1**

1.1 Applications 1
1.2 Limitations of Co-occurrence Data 2
1.3 Overview of Our Approach 3

2. RELATED WORK **5**

3. DATA-DRIVEN SEMANTIC NETWORK **7**

4. INSTANCE REPRESENTATION **9**

5. CONCEPT REPRESENTATION **11**

5.1 Verb Concept Representation 11

6. SHORT TEXT UNDERSTANDING **13**

6.1 Problem Definition 13
6.2 Joint Structured Prediction Model 14
6.2.1 Conditional Model 14
6.2.2 Learning 17
6.3 Experiments 18
6.3.1 Dataset of Labeled Queries 18
6.3.2 Evaluation 18
6.3.3 Results 19

7. OTHER APPLICATIONS **23**

7.1 Word Chunking 23
7.2 Word Sense Disambiguation 23
7.3 Coreference 24
7.4 Triple Extraction 24

8. ACQUIRING INSTANCE-CONCEPT REPRESENTATION **26**

8.1 Diverse Concept Clusters 26
8.2 Vocabulary 27
8.3 Noun Phrase Extraction 27
8.4 Verb Phrase Extraction 28

8.5 Prepositional Phrase Extraction 28
9. CONCLUSION **29**
REFERENCES **30**

ACKNOWLEDGMENTS

I would like to thank my mentor, Prof. Feifei Li, for much advice and tremendous support during my two years at the University of Utah. Feifei navigated me through the research process and let me explore different areas of computer science, which led me to my true passion, artificial intelligence in the field of natural language processing. I am extremely grateful to Feifei for this opportunity.

I would also like to thank Dr. Haixun Wang, who introduce me to the field of natural language processing and gave me the opportunity to work with him. Haixun helped me with the research and gave me much advice on writing and presentation skills.

I would also like to thank my committee members, Prof. Jeff Phillips and Prof. Vivek Srikumar, for much good advice regarding the model and experiments of short text understanding.

CHAPTER 1

INTRODUCTION

Co-occurrence of a word is a distribution over words that frequently occur in the same context. For example, words that frequently co-occurr with the word *apple* include {*orange, banana, iPhone, iPad, CEO, vitamin*}, while frequent co-occurring words for the word *book* are {*title, author, reader, page, ink*} (we omitted the associated probabilities or weights). Co-occurrence is important for modeling language and can be used in many NLP applications.

1.1 Applications

When reading news or other type of text documents, we extensively use the co-occurrence knowledge to understand the meaning of the text. After reading the following sentence, *"xyz is a big metropolitan area with many rich cultural societies"*, we most likely identified *xyz* as a (big) city, because we often see a word (concept) *city* interact with *metropolitan area, society*, and *culture*.

We also use co-occurrence knowledge to figure out how to chunk the words. Given a short text, *"jordan 10 day weather forecast"*, the more likely partition of the text is *"jordan 10 day weather forecast"*, rather than the partition *"jordan 10 day weather forecast"*. The former partition makes more sense since *country* (*Jordan*) occurs more frequently with *weather forecast* than a *basketball shoe* (*jordan 10*) does. We can use the co-occurrence data to assign probabilities to partitions and choose the most likely partition.

Identifying antecedents of mentions in the text is another example where co-occurrence knowledge can be very helpful. In the following sentences, *"The play in the city was very nice. It included many great actors."*, we can infer that the mention *It* refers to the *play*, not the *city*, because the co-occurrence information tells us that plays co-occur more often with actors than cities do.

Understanding short text (e.g. queries, posts) is another example where co-occurrence knowledge is of great importance, since the context of the short text is very limited.

Given a query *"april in paris lyrics"*, we can use co-occurrence knowledge to figure out the proper segmentation and disambiguation of the query, *"april in paris [song] lyrics"*, where *april in paris* and *lyrics* are two segments in the query, and *april in paris* refers to the concept *song*.

There has been a lot of existing work on obtaining word co-occurrence. Brown Corpus co-occurrence [1] obtains the co-occurrences of semantically related pairs of nouns appearing within a window of 250 characters. Topic models (e.g. *LDA*) are another example of word co-occurrence modeling, where each topic is a probability distribution over words that "generate" the topic [2]. Recent work on learning distributed word representations (e.g. *word2vec*) [3, 4, 5, 6] can be also thought of as co-occurrence or context representations. These representations have the property that if words are semantically similar, then the corresponding word vectors should be close (nearby) in the vector space.

1.2 Limitations of Co-occurrence Data

A well-known limitation of co-occurrence data is data sparseness, which is also known as the zero frequency problem [7]. For a majority of pairs, the probability that they co-occur in even a large corpus could be very small. For example, we may observe that the two words *movie* and *premiere* co-ocurr with high frequency. However, for a little known movie x, the probability that x co-occurs with *premiere* is extremely small or even 0 in a large corpus.

A lot of work has been done to address the zero frequency problem. Most such work involves a statistical framework that estimates the probabilities for pairs that were rarely observed or even unobserved in a given sample set [8]. Existing work also tries to reveal the structure or hierarchical data organization for the words and terms, and use such structure for better prediction. For example, Hofmann et al. [9] proposed a novel family of mixture models to explain the observed data by a finite number of shared aspects or clusters.

In this paper, we argue that the major limitation of co-occurrence data is not data sparseness. Rather, the fundamental issue is that co-occurrence data lacks explicit semantics, or in other words, it considers words and terms as atomic units, and no explicit relationships exist among the words and terms. Because of this, we cannot predict the co-occurrence between a rare pair based on the co-occurrences of other pairs that are semantically related to the rare pair. It is not surprising that existing statistical framework, in order to improve prediction, focuses on clustering, grouping, or revealing the internal structure or hierarchical organization of the data. However, statistical modeling often has limited power in revealing the most fundamental structure in the data.

We argue that the most fundamental structure is the type or the conceptual structure in the data. If such knowledge is avalable, we will know that the co-occurrence data of *apple* as a *company* would contribute to co-occurrence data of a concept *company*. Similarly, co-occurrence data of *apple* as a *fruit* would contribute to co-occurrence representation of a concept *fruit*. We can see that type information enables us to better estimate co-occurrence for any pair of words or terms.

Making the type or the conceptual structure in co-occurrence data explicit also enables a wide range of new applications. Currently, the co-occurrence {*orange, banana, iPhone, iPad, CEO, vitamin*} for the word *apple* does not differentiate between concepts *company* and *fruit* that the word *apple* maps to. This limits the use of such co-occurrences or context representations in many NLP applications. One way to solve this problem is to obtain the co-occurrence for a word given a particular concept (sense). In the *apple* example, we would have two co-occurrence lists, one for *apple* as a *fruit* {*orange, banana, vitamin*}, and another for *apple* as a *company* {*iPhone, iPad, CEO*}.

1.3 Overview of Our Approach

Instead of treating words or terms as atomic units, we include structures into co-occurrence data. The structures with which we are concerned contain the following:

- Conceptual structure. We use a large-scale, data-driven semantic network to provide the underlying conceptual structure for the words and terms in our co-occurrence data. This enables us to find not just the co-occurrence information for the word *apple*, but for the word *apple* as a *company* or *apple* as a *fruit*. This also enables us to relate the co-occurrence of *apple* and *google*, as well as co-occurrence of *apple* and *banana*. Thus, it makes it easier to reason or inference co-occurrence probabilities with the underlying conceptual system.

- Syntactic structure. We explore co-occurrence of two terms on the dependency tree. For example, *apple* as a *company* may co-occur with verb phrases such as {*produce, design, invest in*} (either as subject or direct object), while *apple* as a *fruit*, co-occurrs with verb phrases such as {*eat, digest, keep doctor away*} (either as subject or direct object). Similarly, we may collect co-occurring prepositional phrases for a word or term as well. This strong typed co-occurrence information is valuable to many applications.

In summary, the novelty of our work can be summarized as follows:

- We introduce term-concept co-occurrence, where the co-occurrence is obtained for a term given its concepts.
- We present an approach to learn concept co-occurrence from term-concept co-occurrence.
- We obtain co-occurrences between noun phrases, not just single words.
- Our co-occurrences include verb phrases and prepositional phrases.
- We present a joint structured prediction model for short text understanding and extensively evaluate the performance of concept co-occurrence vs. co-occurrence alone.
- We present several NLP applications, where concept aware co-occurrence can be naturally applied and potentially improve the performance of the system.

CHAPTER 2

RELATED WORK

Most of the existing work has been on term- or word-based representations and not concept-based. Examples of work in this area are Brown Corpus co-occurrence [1], where the author searched for co-occurrences of semantically related pairs of concrete nouns appearing within a window of 250 characters. Another interesting work is co-occurrence of antonym adjectives (big-little) [10], where they showed that adjectives tend to occur in the same sentence as their antonyms far more frequently than expected by chance.

Topic models (e.g. *LDA*) "cluster" words that tend to co-occur into the same topic, and can be thought of as a type of co-occurrence representation for the topics. There has been extensive research in this area and some of the work uses existing concept knowledge to improve topic models [2], but we were unable to find topical models that deal with concepts exclusively. Additionally, it is hard to interpret or label the learned topics (kind of concepts).

Recently, there has been a lot of work on learning distributed word representations [3, 4, 5, 6]. The idea is to learn dense vectors for words, such that, if words are semantically similar, then corresponding word vectors should be close (nearby) in the vector space. The representations are learned for words (and some word phrases), but during learning there is no injection of the concept knowledge about the word, resulting in a word representation that is fairly insensitive to the concepts (senses) the word possesses.

Another major drawback of much existing work is that representations or co-occurrences are between single words, and not phrases. We do not have the representation for phrases, such as *Bruno Mars, San Francisco, financial institution*, or *weather forecast*. Word phrases are very important to obtain holistic representations. Additionally, they introduce specificity and decrease the amount of ambiguity in the representations: for the term *bank*, it is much more informative to know that it co-occurs with *financial institution* than simply *institution*.

Moreover, the representation between concepts and noun phrases is not sufficient. Our representations include verb phrases and prepositional phrases. For a given concept, what are the typical verb phrases that interact with this concept, or, for a given verb phrase, what are the typical concepts that interact with this verb phrase? Similarly, we have a list of frequently used prepositional phrases with the concepts.

Some approaches rely on *WordNet* dataset, where among other information, each word has a set of distinct *synsets* (concepts). The two major drawbacks with *WordNet* compared to our *IsA network* is that *WordNet* does not have many proper nouns and there is no typicality scores for the words. Proper nouns are very important, especially for learning concepts related to people. The typicality is crucial for concept learning [11], since some instances are much better prototypes for concepts than others (e.g. *sofa* is a good example of concept *furniture*, while *car char* is not).

CHAPTER 3

DATA-DRIVEN SEMANTIC NETWORK

In our work, we create concept aware co-occurrence data, and the concepts in our data come from a data-driven semantic network. Nodes in the semantic network are words or terms (multiword expressions). Edges in the semantic network denote relationships among the nodes. In our work, we are mostly concerned with the isA relationship, also known as the hyponym-hypernym (instance-concept) relationship. Each isA relation is also associated with conditional probabilities known as typicality scores. Formally, an *IsA relation* is a tuple of the form $(i, c, p(i|c), p(c|i))$, where i is a **hyponym** (instance), c is a **hypernym** (concept), $p(i|c)$ is the typicality of an instance i given the concept c, and $p(c|i)$ is the typicality of concept c given the instance i. The typicality scores are important for inferencing, because, for example, not every instance in a concept is equal. When someone mentions *bird*, it is more likely we think of *robin* than *penguin*. This is captured by the score $p(robin|bird) > p(penguin|bird)$.

The isA relationship is important because it enables generalization, which lies at the core of human cognition. Because of this, the isA relationship is the backbone of almost every taxonomy, ontology, and semantic network. To understand its significance, consider two terms a and b, and $R(a, b)$, which denotes a certain relationship between a and b. It can be shown that the isA relationship may be used to generalize R, or in other words, it may enable us to know more about R. For example, if we know b' is a hypernym or hyponym of b, chances are $R(a, b')$ also holds. In our case, R may denote the co-occurrence relationship, that is, $R(a, b)$ holds if a frequently co-occurs with b. For instance, we may have R(`premiere`, `movie`). Now, given that we know (`x`, `movie`) is an isA pair, therefore, x is a movie. No matter how little known x is, we may infer R(x, `movie`) from that of R(`premiere`, `movie`).

We obtain the *IsA network* by extracting the hypernym-hyponym relationships from sentences containing *Hearst pattern* (e.g. I have visited countries such as China, Japan, and South Korea $\Rightarrow$ *country* is concept with instances *China, Japan*, and *South Korea*).

There has been a lot of work on extracting high-quality *IsA network* from text data and it is beyond this work. Examples of such datasets are Google ConceptNet and Microsoft Probase [12]. Google Concept Net contains over 60 million nodes (hypernyms, hyponyms) and over 140 million edges (hypernym-hyponyms relations).

CHAPTER 4

INSTANCE REPRESENTATION

Before we dive into instance representation, it is vital to understand that whenever we refer to an instance, we always mean an instance of a certain concept. In other words, instances and concepts are inseparable – we can not refer to an instance *apple* without specifying one of its concepts, *company* or *fruit*. Formally, we are given a collection of instance-concept pairs (i, c), where i is an instance and c is its concept (e.g. $(apple, company), (apple, fruit)$). *IsA network*, described in the previous chapter, provides instance-concept pairs, along with their typicality scores.

In our work, we decided to represent an instance with the context in which the instance occurs. The context is extracted only within the same sentence and consists of co-occurring noun, verb, and prepositional phrases.

Let our instance-concept pair be (*apple*, *company*) and the sentence be *"iPhones produced by Apple are great mobile phones"*. Then we extract the following co-occurring noun phrases {*iPhones, mobile phones*}, verb phrase subject {}, verb phrase object {*produced by*}, and preposition phrases {*by*} with instance *apple* of a concept *company*.

Formally, the representation of instance-concept pair (i, c), denoted as $IR(i, c)$ is a tuple with four components:

- $NP(i, c)$ is a probability distribution over co-occurring noun phrases with instance i of a concept c.
- $VP_{sub}(i, c)$ is a probability distribution over co-occurring verb phrases where instance i of a concept c is a subject of the verb phrase.
- $VP_{obj}(i, c)$ is a probability distribution over co-occurring verb phrases where instance i of a concept c is a direct object of the verb phrase.
- $PP(i, c)$ is a probability distribution over co-occurring prepositional phrases where instance i of a concept c is a prepositional object of the preposition phrase.

We can obtain the instance representations by extracting the co-occurring noun, verb, and prepositional phrases from sentences from billions of documents. We explain the procedure in detail in Chapter 8.

CHAPTER 5

CONCEPT REPRESENTATION

Each concept represents a set of instances and the instances describe or define the concept. We are going to use instance-concept relationships, given by *IsA network*, and instance representations to build a concept representation.

Let us assume the concept c is *fruit* and the instances are {*orange, apple, tomato, olive*}, with the corresponding typicalities $p(orange|fruit)$, $p(apple|fruit)$, $p(tomato|fruit)$, and $p(olive|fruit)$. To obtain the concept representation for *fruit*, denoted by $CR(fruit)$, we compute the following weighted sum:

$$\begin{aligned} CR(fruit) \propto\ & p(orange|fruit)IR(orange, fruit) \\ & + p(apple|fruit)IR(apple, fruit) \\ & + p(tomato|fruit)IR(tomato, fruit) \\ & + p(olive|fruit)IR(olive, fruit). \end{aligned}$$

We used the representation of instances (IR) to build a representation of a concept. In the process, we consider how typical instances are for the concept, which aligns with the concept theory that was introduced in the previous chapters.

Formally, concept representation $CR(c)$ of a concept c is computed as follows:

$$CR(c) \propto \sum_i p(i|c)IR(i, c),$$

where $p(i|c)$ is the typicality score of an instance i being a concept c and $IR(i, c)$ is the instance representation of instance i of a concept c. The weight $p(i|c)$ is applied to every individual component (NP, VP_{sub}, VP_{obj}, PP) of the IR tuple.

5.1 Verb Concept Representation

We have talked about the representation of a concept, where the concept representation consists of co-occurring noun phrases, verb phrases, and prepositional phrases. Now, let

us take a look at verb phrase representation, which consists of co-occurring concepts for a given verb phrase.

For a verb phrase *watch*, a list of co-occurring subject concepts is {*Person, Child, Student, Move Expert*}, and a list of co-occurring object concepts is {*TV Show, Movie, Documentary, Game*}. Verb phrase concept representation gives us the concepts that often interact with verb phrase as a subject or as an object. It is a distribution over concepts for subject and object arguments of a verb phrase.

Formally, verb concept representation $VR(v)$ of a verb phrase v is a tuple with two components:

- $VR_{sub}(v)$ is a prob. distribution over concepts that are subjects of a verb phrase v.
- $VR_{obj}(v)$ is a prob. distribution over concepts that are objects of a verb phrase v.

We can obtain verb concept representation by using existing concept representation. Remember that $CR(c)$ contains $VP_{sub}(c)$, distribution over verb phrases where c is a subject of a verb phrase, and $VP_{obj}(c)$, distribution over verb phrases where c is an object of a verb phrase. We can compute the verb concept representation $VR(v)$ as follows:

$$VR_{sub}(v) \propto \sum_{c} p(c) VP_{sub}(c, v),$$

$$VR_{obj}(v) \propto \sum_{c} p(c) VP_{obj}(c, v),$$

where $p(c)$ is a probability of a concept c (from *IsA network*), $VP_{sub}(c, v)$ is a probability of a verb phrase v given a concept c as subject, and similarly for $VP_{obj}(c, v)$.

CHAPTER 6

SHORT TEXT UNDERSTANDING

In this chapter, we present how can we apply concept aware co-occurrence to short text understanding. Short text usually does not have the syntax of a normally written language. The context information within the short text is very limited due to its conciseness and shortness. In recent years, short text has become ubiquitous: we use queries (short text) to communicate our requests to a search-engine; writing and reading posts (short text) on social networking websites is another example. Thus, understanding of short text has tremendous value for the users and service providers.

By short text understanding we refer to determining the proper segmentation of the short text, and disambiguation of the segments into their proper senses (concepts). Intensive knowledge processing techniques are crucial for successful understanding of short text [13]. Our approach uses semantic network coupled with concept aware co-occurrence knowledge to perform short text understanding. We propose a joint structured prediction model that leverages this knowledge to compute the most likely understanding of the short text. The purpose of this chapter is not to compare our system to other systems, but to show that concept co-occurrence can boost the performance of short text understanding over co-occurrence data alone.

6.1 Problem Definition

In this section, we formally define the problem of short text understanding. The input is a sequence of tokens $T = t_1, \ldots, t_N$ that represents the short text. Examples of input are: *april in paris lryics* and *harry potter watch*. The task of short text understanding consists of two subproblems:

- **Segmentation** of the input tokens into meaningful segments $S = s_1, \ldots, s_n$.
- **Disambiguation** of the segments into their proper senses (concepts) $C = c_1, \ldots, c_n$.

Thus, the output of short text understanding consists of segmentation and disambiguation assignments to the input tokens.

For the first example, a possible understanding or interpretation of the query could be:

$$\underline{\text{april in paris}}\ [\text{song}]\ \underline{\text{lyrics}}\ [\text{music}],$$

where the segments are $s_1 = \underline{\text{april in paris}}$, $s_2 = \underline{\text{lryics}}$, and the disambiguated concepts are $c_1 = \textit{song}$, $c_2 = \textit{music}$. A possible interpretation for the second example can be:

$$\underline{\text{harry potter}}\ [\text{brand}]\ \underline{\text{watch}}\ [\text{accessory}],$$

where the segments are $s_1 = \underline{\text{harry potter}}$, $s_2 = \underline{\text{watch}}$, and the disambiguated concepts are $c_1 = \textit{brand}$, $c_2 = \textit{accessory}$.

6.2 Joint Structured Prediction Model

Segmentation and disambiguation are very related problems - in many cases to do a correct segmentation, we need to disambiguate the phrases at the same time. This motivated us to design a joint structured prediction model for short text understanding. The output of the model is a joint assignment of segmentation and disambiguation to the input short text.

Since short text consists of a small number of tokens, we can afford to enumerate all possible segmentations. To enumerate all possible segmentations of a short text of length n, it can take exponential time and space in n. However, queries are very short and rarely consist of 10 or more tokens, thus typically $n < 10$. Moreover, since we operate with a fixed and precomputed vocabulary of phrases, we do not need to enumerate all the possible segmentations, but only all the segmentations in which segments belong to the vocabulary. In practice, this significantly reduces the number of all possible segmentations.

For a particular segmentation, we exhaust all the possible concept assignments to the segmentation. In Chapter 8, we explain how we group similar concepts to diverse concept clusters, which gives us a very small number of different concept clusters for every segment, thus exhausting all the possible concept assignments is not expensive.

At this point, we can enumerate all the possible segmentations and concept assignments: at each step, we obtain a segmentation S and concept assignment C for the given input tokens T.

6.2.1 Conditional Model

This section introduces conditional models that leverage the co-occurrence data to compute the probability of a given short text understanding.

We denote $Y = \{(s_1, c_1), \ldots, (s_n, c_n)\}$ as the output by simply pairing the corresponding segments and concepts. We also denote $Y_{-i} = Y \setminus \{(s_i, c_i)\}$ as the output without the i-th segment and concept. All of our conditional models are exponential models and have the following form:

$$P(Y|T) \propto \exp\left[\Phi(Y, T)\right],$$

where $\Phi(Y, T)$ is function that scores the output Y for the input tokens T. In all cases, the output of short text understanding is generated by maximizing the score function $\Phi(Y, T)$ over Y (as described in the previous section). In the sections that follow, we will define a serious of models of increasing expressiveness / complexity by defining different scoring functions $\Phi(Y, T)$ over Y.

6.2.1.1 Simple Independent Model

Let us first take a look at a very simple model that assumes mutual independence between tokens and pairs of segment-concept. Function $\Phi(Y, T)$ factorizes as follows:

$$\begin{aligned}
\Phi(Y, T) &= \Phi(t_1, \ldots, t_N, (s_1, c_2), \ldots, (s_n, c_n)) \\
&= \sum_{i=1}^{N} \hat{\Phi}_S(t_i, (s_1, c_2), \ldots, (s_n, c_n)) \\
&= \sum_{i=1}^{N} \sum_{j=1}^{n} \Phi_S(t_i, s_j, c_j)
\end{aligned}$$

Such independence assumption typically does not hold in real-world queries; however, some kind of factorization is necessary, since we do not have and cannot practically obtain multivariate joint scoring functions.

More importantly, the score $\Phi_S(t_i, s_j, c_j)$ depends only on the individual tokens, rather than on the other segment and concepts pairs Y_{-j}. This is a problem, because other segments and concepts, Y_{-j}, do not affect the score of the current segment and concept pair (s_j, c_j).

Example: Let the query be *bruno mars the lazy song*, where the proper segmentation is *bruno mars* and *the lazy song*. The simple independent model would not consider the two proper segments, but rather individual tokens {*bruno, mars, the, lazy, song*}, when computing the score $\Phi_S(t_i, s_j, c_j)$. In addition, it also does not take into consideration the concepts of the other segments, which can significantly help with short text understanding: concept *singer* (from segment *bruno mars* often co-occurs with concept *song* (from segment *the lazy song*), thus co-occurrences between concepts can be very helpful.

What we really need is a model that takes into consideration the rest of the context (other segments and concepts). We define such a model in the next section.

6.2.1.2 Global Model

In this section, we define a model that addresses the problems introduced in the previous section. The global model is defined as follows:

$$\Phi(Y,T) = \sum_{i=1}^{n} \hat{\Phi}_G(Y_{-i}, s_i, c_i, T)$$

Term $\hat{\Phi}_G(Y_{-i}, s_i, c_i, T)$ denotes the score of the rest of the output Y_{-i}, given the i-th segment and concept. Intuitively we can think of this term as how well the i-th segment and concept pair (s_i, c_i) correlates or generates the other segment and concepts pairs, denoted by Y_{-i}. The scoring function $\hat{\Phi}_G(Y_{-i}, s_i, c_i, T)$ is a complex scoring function of many variables. Thus, we need to further decompose the scoring function as follows:

$$\hat{\Phi}_G(Y_{-i}, s_i, c_i, T) = \sum_{j=1\ i\neq j}^{n} \Phi_G(s_j, c_j, s_i, c_i, T)$$

Function $\Phi_G(s_j, c_j, s_i, c_i, T)$ is the *pairwise score* of i-th and j-th segment-concept pair. We can think of pairwise score as as the strength of correlation between i-th and j-th segment-concept pair.

Note that the model does not directly operate with tokens, but rather with segments (sequence of tokens), which alleviates the independence assumption between tokens made in the previous model. The model also uses the concept information of every other segment. This solves many problems with the *Simple Independent Model* from the previous section, and give us a global model that takes into account all of the context.

We have left to define the pairwise score $\Phi_G(s_j, c_j, s_i, c_i, T)$. In the next two sections, we define two different types of scoring functions Φ_G, depending on the type of data we use: co-occurrence or concept co-occurrence.

6.2.1.3 Co-occurrence Model

The first model assumes that the pairwise score decomposes into a scoring function between a pair of segments, and a scoring function between a concept and a segment. This gives the following scoring function:

$$\Phi_G(s_j, c_j, s_i, c_i, T) = \Phi_{G_{1ss}}(s_j, s_i) + \Phi_{G_{1cs}}(c_j, s_i)$$

We can think of score $\Phi_{G_{1ss}}(s_j, s_i)$ as the correlation between segment s_j and segment s_i. Score $\Phi_{G_{1cs}}(c_j, s_i)$ is the correlation between concept c_j and segment s_i.

6.2.1.4 Concept Co-occurrence Model

The second model assumes that the pairwise score decomposes into a scoring function between two segments and a concept, a scoring function between segment and concept, and a scoring function between pair of concepts. This gives the following model:

$$\Phi_G(s_j, c_j, s_i, c_i, T) = \Phi_{G_{2ssc}}(s_j, s_i, c_i) + \Phi_{G_{2sc}}(s_j, c_i) + \Phi_{G_{2cc}}(c_j, c_i)$$

6.2.2 Learning

We have defined three different models that score the output Y given the input tokens T. As mentioned before, these models require joint inference across all the segmentations and concept assignments. However, *joint learning* is infeasible because there is no labeled data. Thus, we propose to use a conditional constraint model framework that admits decomposed learning and still allows joint inference. We define the scoring functions using co-occurrence data in Table 6.1.

Term $TC(t_i, t_j)$ denotes the term-term co-occurrence probability between terms t_i and t_j. This is a standard term-term co-occurrence without any information about the concepts. Note that the co-occurrence model defined in the previous section is unable to use the provided concept information, since term-term co-occurrence data is not concept aware. Both scoring functions $\Phi_{G_{1ss}}(s_j, s_i)$ and $\Phi_{G_{1cs}}(c_j, s_i)$ treat the input arguments as terms, regardless of whether the input arguments are concepts.

In the concept co-occurrence model, we are directly leveraging instance-concept co-occurrence data $\Phi_{G_{2ssc}}(s_j, s_i, c_i)$, which has a very different co-occurrence distribution if we change the concept of a given instance (co-occurrence for apple as fruit is much different from co-occurrence for apple as a company). This brings tremendous improvement to the model. We also use concept co-occurrence weight $\Phi_{G_{2sc}}(s_j, c_i)$ between concept and segment, which directly deals with the co-occurrence sparsity problem that was introduced previously. In addition, we also use the concept co-occurrence between the concepts themselves $\Phi_{G_{2cc}}(c_j, c_i)$,

Table 6.1. Definitions of scoring functions in terms of co-occurrence.

Scoring function	Co-occurrence
$\Phi_{G_{1ss}}(s_j, s_i)$	$\log(TC(s_j, s_i))$
$\Phi_{G_{1cs}}(c_j, s_i)$	$\log(TC(c_j, s_i))$
$\Phi_{G_{2ssc}}(s_j, s_i, c_i)$	$\log(IR(s_i, c_i).NP[s_j])$
$\Phi_{G_{2sc}}(s_j, c_i)$	$\log(CR(c_i).NP[s_j])$
$\Phi_{G_{2cc}}(c_j, c_i)$	$\log(CR(c_i).NP[c_j])$

which brings additional signal into the model. Note that notation $IR(s_i, c_i).NP[s_j]$ denotes the instance co-occurrence probability of term (noun phrase) s_j with instance s_i as a concept c_i. Similarly, $CR(c_i).NP[s_j]$ denotes the concept co-occurrence probability of term s_j (noun phrase) with a concept c_i.

The models that we defined use solely co-occurrences between noun phrases. To fully leverage the concept co-occurrence knowledge, we can extend the models to use co-occurring verb and prepositional phrases.

6.3 Experiments

In this section, we present the comparison of the co-occurrence model vs. the concept co-occurrence model on a set of labeled web-search queries.

6.3.1 Dataset of Labeled Queries

Since there is no publicly available benchmark or dataset of labeled queries that we know about, we generated a dataset of labeled queries ourselves. By labeled dataset we mean that we have the information about the proper segmentation and disambiguation (concept assignment) of every query in the dataset. We present an approach that leverages the *IsA network* to generate labeled queries.

To do this effectively, we define a **query pattern**, which is simply a sequence of words (tokens) and concepts. An example would be *[song] lyrics*, where *song* is a concept and *lyrics* is a token. We can have multiple concepts or tokens intermixed in a single query pattern. Given such a query pattern, we sample instances of concepts from the query pattern using *IsA network*, and then replace the concepts with sampled instances to obtain a concrete query. Let us say we sample {*april in paris, grenade, the way we were*} instances of a concept *song*, which give us the following concrete queries: *april in paris lyrics*, *grenade lyrics*, and *the way we were lyrics*. This generative process gives us labeled queries: we know the proper segmentation of the query as well as the concept assignment to segments.

More examples of query patterns: *[movie] soundtrack*, *[fruit] vitamin*, *[food] recipe*, *[watch brand] watch*, *watch [tv show]*, *read [book]*.

6.3.2 Evaluation

In this section, we define two metrics to measure the similarity between two labeled queries. The metrics will be used for evaluation and comparison of our models.

Exact Metric is a simple binary metric and requires both labeled queries, q_1 and q_2, to be exactly the same in order to be one; otherwise the metric is zero. This is a very strict

metric: a single mistake, either with segmentation or concept assignment, and the entire result will receive a score of zero. Formally, we can define the metric as follows:

$$em(q_1, q_2) = \begin{cases} 1 & q_1, q_2 \text{ same segments and concepts} \\ 0 & \text{otherwise} \end{cases}$$

Rank Metric is a more forgiving version of the exact metric in a sense that it does not require the exact matching of the concepts in the labeled queries. The idea is that the output of the short text understanding is a ranking over the concepts for segments in short text (so far we had just a single concept for every segment). This enables us to compute the rank of the labeled concepts in the output.

In the case of query *april in paris lyrics*, the system would output a concept ranking { *(1) song, (2) hit, (3) book* } for segment *april in paris*.

Returning a ranking over the concepts is not a problem for short text, since we can afford to do an exhaustive search over all the possible concepts assignments as described in the previous section. The metric still requires for the queries to have the same segmentation, otherwise it is zero. Note that we can rank the segmentations as well, but we avoided this for the sake of simplicity of the metric.

The intuition behind the metric is that the concept with which we expect the segment to be labeled should be as high as possible on the concept ranking for that segment. Let us say that we expect the concept to be *hit* for segment *april in paris*. If our method produced the ranking that is given above, then the score would be $\frac{1}{2}$ for this query, since the expected concept *hit* is in second place in the ranking. As you can see, we use inverse rank instead of rank, since it is a much more robust measure and it is not sensitive to outliers. We denote the rank metric of two labeled queries q_1 and q_2 as $rm(q_1, q_2)$.

6.3.3 Results

In this section, we present the empirical results of the co-occurrence model vs. concept co-occurrence model using the exact and rank metric on the dataset of labeled queries that we generated.

The labeled query dataset consists of 15 query patterns, and for each of them we generated 100 concrete queries, resulting in 1500 queries. We show some of the query patterns and query examples that we used in our experiments in Table 6.2.

6.3.3.1 Overall Performance

Table 6.3 shows the overall performance of co-occurrence vs. concept co-occurrence model. We can see that concept co-occurrence is strongly superior over co-occurrence and

Table 6.2. Query patterns and query examples.

Query Pattern	Example
[song] lyrics	april in paris lryics
[song] sheet	goldfinger sheet
[movie] premier	james bond premiere
[watch brand] watch	rolex watch
[food] recipe	cake recipe
[fruit] vitamin	apple vitamin

wins by a large margin. This means that concept co-occurrence data can significantly help with short text understanding task. The results for exact and rank metric are averaged over all the query examples in our dataset.

In the case of exact metric, we can see that concept co-occurrence has over 15% improvement over co-occurrence. Co-occurrence was able to correctly predict about 48% of the queries, while concept co-occurrence predicted correctly over 63% of the queries. This is a significant improvement and can make a big difference when running in real-world environments.

With rank metric, co-occurrence achieved a result of 0.597, which means that on average, the correctly predicted concept was in second place in the ranking. Concept co-occurrence achieved over 0.12 improvement, resulting in 0.718 average rank metric.

6.3.3.2 Easy Examples

When running the experiments, we found that there are certain query patterns for which co-occurrence does surprisingly well - similar to concept co-occurrence (Table 6.4). Examples of such queries are: *[song] lyrics, [song] music video, [university] degree.* A detailed look into co-occurrence showed that instances (*april in paris*) of such concepts (*[song]*) highly correlate with certain very specific words (lyrics), which is the reason the co-occurrence model performs very well. In other words, when we are dealing with unambiguous context, co-occurrence seems to perform well. However, concept co-occurrence still outperforms the co-occurrence by at least several percentage points on the easy example as well.

6.3.3.3 Hard Examples

We have also found that certain query patterns are very hard and co-occurrence performs significantly worse than concept co-occurrence (Table 6.4). An example of a hard query is *[song] sheet.* The reason for this is because the context is extremely ambiguous. The

second reason is sparsity: many words will not co-occur with atypically words, but when used together, the meaning of the words is clear.

In our example *[song] sheet*, the context word is *sheet*, which has many different meanings depending on the context in which it occurs. In the context of a *song*, the word *sheet* refers to *music sheet*. Many songs will not co-occur with *sheet* or *music sheet* when obtaining co-occurrence data. Concept co-occurrence naturally takes care of the sparsity problem through the generalization process.

Table 6.3. Overall performance of the methods.

Model	Exact Metric	Rank Metric
Co-occurrence	0.479	0.597
Concept co-occurrence	0.633	0.718

Table 6.4. Rank metric for a subset of query patterns.

Query Pattern	Co-occ.	Concept co-occ.
[song] lyrics	0.801	0.895
[song] sheet	0.305	0.681
[film] premiere	0.776	0.873
[watch] watch	0.173	0.59
[food] recipe	0.473	0.592
[furniture] design	0.557	0.651

CHAPTER 7

OTHER APPLICATIONS

In this chapter, we present other applications where we see that concept co-occurrence can be applied to improve the performance. We will focus on word chunking, word sense disambiguation, coreference resolution, and triple extraction.

7.1 Word Chunking

Word chunking is a problem of partitioning a given sequence of tokens (e.g. sentence) into contiguous spans of tokens, called chunks of groups. Each chunk should represent a meaningful sequence of tokens given the entire sequence. Chunks are often named entities or word phrases representing concepts.

Let us take a look at a few examples of correct word chunking of a sequence of tokens:

1. April in Paris lyrics
2. vacation April Paris

Analyzing the above examples, we can quickly see that *April in Paris* is a name of a song and *lyrics* often refers to the song concept. Similarly, *vacation* is a concept that very often occurs with time (*April*) and place (*Paris*).

The problem of word chunking comes down to what is a reasonable or coherent partition of a given sequence of tokens. In many cases, including the example above, context knowledge about instances and concepts is required to resolve the correct word chunking. Exactly such type of knowledge is stored in concept co-occurrence, since it contains context information about the instances and concepts.

7.2 Word Sense Disambiguation

Word sense disambiguation (WSD) is about determining which sense of the a word or word phrase is activated by its use in a particular context. Formally, we are given a

sequence of words $\{w_1, w_2, \ldots, w_n\}$, and the task is to assign appropriate senses $S(w_i)$ to every word w_i, where $S(w_i) \subseteq D(w_i)$ and $D(w_i)$ is a set of all senses for word w_i.

Let us take a look at two examples of WSD:

1. watch Harry Potter[movie]
2. read Harry Potter[book]

In the first example, *Harry Potter* is a movie based on the context word *watch*, while in the second example, the word *read* signalizes that *Harry Potter* stands for a book.

Using contextual noun phrases and verb phrases is often enough to identify the correct sense of the word. Concept co-occurrence that we introduced includes co-occurring noun, verb, and prepositional phrases, in addition to being concept aware.

7.3 Coreference

In a **coreference** problem, we would like to identify to what the mentions in the text refer. It is easy to find examples where a coreference problem is very hard and our representations are clearly not sufficient to solve the problem. We limit ourselves to a subset of **nominal coreference**. Below is an example of nominal coreference:

Google invested in Magic Leap. The Mountain View giant is planning to use it for Google Glass project.

The task of coreference resolution is to figure out that *Mountain View giant* refers to *Google*. Concept co-occurrence would give us a much higher weight of *Google* as *company* co-occurring with *Mountain View giant*, than with any other instance of the concept *company* (*Magic Leap*). Thus, concept co-occurrence can be a strong signal into a corefrence resolution system.

7.4 Triple Extraction

In **triple extraction**, our goal is to find a verb phrase and two (or at least one) of its argument, subject, and object. We presented the verb concept representation, which indicates what kind of concepts often occur as a subject or an object with a given verb phrase. This information can be used to determine the likelihood of an argument candidate being an argument of a verb phrase. Let us take a look at the following example:

Students often watch online lectures while eating pizza.

Let us assume that we have two candidates, *lectures* and *pizza*, for an object argument for a verb *watch*. The verb concept representation tells us that verb *watch* much more often occurs with concept *lecture* than concept *food*, thus we can safely choose the *lecture* as the object argument.

CHAPTER 8

ACQUIRING INSTANCE-CONCEPT REPRESENTATION

In this chapter, we describe how can we efficiently and effectively obtain instance-concept representation $IR(i, c)$, for instance i of a concept c. Before we do that, we introduce concept clusters: instead of obtaining the representation for every (i, c) pair in our *IsA network*, we only obtain the representation for every (i, CC_s), where CC_s is a concept cluster representing similar concepts for instance i.

8.1 Diverse Concept Clusters

IsA network stores the relationships between concepts and instances: for any instance i, we have a set of concepts C_i. For large *IsA networks*, especially when obtained in a data-driven approach, the set C_i can be very large, containing tens or hundreds of concepts for an instance i. Many of these concepts are very similar (instance *apple* contains the concepts *company*, *it-company*, *phone-company*) and we may want to group them into a **concept cluster**. There are applications where differentiating between fine-grained concepts is important, but for our purposes, we focused on obtained concept representations for diverse concepts.

Our goal is to select a set of diverse concept clusters, where each cluster contains similar concepts, and clusters themselves are diverse. For instance *apple*, we would select the following two diverse concept clusters: {*company, it-company, phone-company, tech giant*} and {*food, fruit, fresh-fruit*}. Note that concept clusters are instance specific - instances can have different concept clusters. For each concept cluster, we also select the representative of the cluster, which can be used as an id for that cluster.

We employed two main techniques for grouping two concepts into one concept cluster:

- If the concepts have the same head word (*it-company, company*).
- If the concepts share more then 50% of the instances.

Diverse concept clusters are an input to our method of extracting instance-concept representations (see next section). Formally, for every instance i, we store a set of concept clusters $\{CC^i_{c^i_1}, \ldots CC^i_{c^i_n}\}$, where $\{c^i_1, \ldots, c^i_n\}$ are concept cluster representatives.

8.2 Vocabulary

Our instance-concept representation consists of co-occurring noun, verb, and prepositional phrases. For each of these three categories, we define the vocabulary of possible strings:

- **Noun phrases:** a set of instances and concepts from *IsA network* (e.g. *apple, company, fruit, April in Paris, New York, I love you*).
- **Verb phrases:** verb phrases that occur at least 1000 times on the web (e.g. *play, influenced by, ask congress, check playlist*).
- **Prepositional phrases:** a set of all English prepositions and prepositional phrases found in the English dictionary (e.g. *alongside, due to, as far as, with regard to*).

8.3 Noun Phrase Extraction

Noun phrase extraction deals with extracting co-occurring noun phrases with an instance-concept pair (i, c). In this context, concept c is the representative concept for the concept cluster CC^i_c as described in previous section. Now, given a particular sentence containing instance i, how do we determine the sense or concept c for an instance i within the sentence. We employ the following simple, but very effective and efficient, method:

We declare i is of concept c (in a particular sentence) if i occurs and $c' \in CC^i_c$ occurs and none of $d \in CC^i_e, e \neq c$ occur. Otherwise, we ignore the sentence for instance i.

In other words, we only declare instance i of being of a concept c (in a particular sentence) if there is no presence of concepts from other concept clusters.

To illustrate this, let us look at an example from the previous section where our instance was *apple* and concept clusters were {*company, it-company, phone-company, tech giant*} and {*food, fruit, fresh-fruit*} with representatives *company* and *fruit*, respectively. We are given two sentences:

1. *Silicon Valley company Apple produces great phones.*

2. *Many companies are involved in fruit business of apple juice.*

In the first sentence, there is a concept (*company*) from the first concept cluster, and there are no concepts from the second concept cluster; therefore, we declare *apple* is of concept *company*. In the second sentence, concept (*company*) is from the first concept cluster, while concept *fruit* is from the second concept cluster; we have interference of concept clusters and therefore, we ignore the sentence.

This techniques is fairly simple, but very effective and works very well in practice. Since we throw away every sentence where we are not "sure" about the concept, the method may not a have high recall, but it has a very high precision. In the presence of big text data, we can afford to throw away many sentences and still obtain very accurate instance-concept co-occurrence statistics.

Co-occurring noun phrases are simply all the noun-phrases in our vocabulary that occur in the same sentence with the identified instance-concept pair (i, c). We obtain all such co-occurrence with their counts and transform them into a probability distributions to obtain $NP(i, c)$.

8.4 Verb Phrase Extraction

In verb phrase extraction, we find co-occurring verb phrases with an instance-concept pair (i, c). Note that we differentiate situations, where instance-concept pair is an object or subject of a verb phrase. To do this, we used a triple extractor that returns triple relations of the form *(subject, verb phrase, object)* within the sentence. We used the same method, as described in the previous section, to identify concept c for *subject* and concept c' for *object*, obtaining the co-occurrence between the *verb phrase* and $(subject, c)$ and the co-occurrence between the *verb phrase* and $(object, c')$. We again count all such co-occurrences and transform them into a probability distribution to obtain $VP_{sub}(i, c)$ and $VP_{obj}(i, c')$

8.5 Prepositional Phrase Extraction

Prepositional phrase extraction finds co-occurring prepositional phrases with an instance-concept pair (i, c). To do this, we searched for *pobj* relation between prepositional phrase *pp* and prepositional object i in dependency tree of the sentence. Then, we identify the concept c for the prepositional object i using the method described above and output the co-occurrence between *pp* and (i, c) We count all such co-occurrences and transform them into a probability distribution to obtain $PP(i, c)$.

CHAPTER 9

CONCLUSION

The major limitation of term co-occurrence is that there is no explicit conceptual or syntactical relationship between terms. This motivated us to define concept co-occurrence that contains co-occurring noun, verb, and prepositional phrases, and is explicitly related to concepts. We showed how can we successfully obtain high-quality instance-concept co-occurrence using large corpus of sentences. Concept representations can be obtained from instance representations through the process of generalization, which is facilitated by *IsA network*.

We applied concept aware co-occurrence to the problem of short text understanding. We defined a joint structured conditional model that leverages concept co-occurrence knowledge to compute the proper segmentation and disambiguation of the query. Extensive evaluation showed that concept co-occurrence is superior over co-occurrence. We also outlined several other NLP applications that could benefit greatly with the use of concept co-occurrence.

REFERENCES

[1] D. P. Spence and K. C. Owens, "Lexical co-occurrence and association strength," *Journal of Psycholinguistic Research*, vol. 19, no. 5, pp. 317–330, 1990.

[2] C. Chemudugunta, P. Smyth, and M. Steyvers, "Combining concept hierarchies and statistical topic models," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, ser. CIKM '08. New York, NY, USA: ACM, 2008, pp. 1469–1470. [Online]. Available: http://doi.acm.org/10.1145/1458082.1458337

[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[4] R. JeffreyPennington and C. Manning, "Glove: Global vectors for word representation."

[5] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in Neural Information Processing Systems*, 2014, pp. 2177–2185.

[6] O. Levy, Y. Goldberg, I. Dagan, and I. Ramat-Gan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, 2015.

[7] I. H. Witten and T. Bell, "The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression," *Information Theory, IEEE Transactions on*, vol. 37, no. 4, pp. 1085–1094, 1991.

[8] P. D. Turney, P. Pantel *et al.*, "From frequency to meaning: Vector space models of semantics," *Journal of artificial intelligence research*, vol. 37, no. 1, pp. 141–188, 2010.

[9] T. Hofmann and J. Puzicha, "Statistical models for co-occurrence data," 1998.

[10] J. S. Justeson and S. M. Katz, "Co-occurrences of antonymous adjectives and their contexts," *Computational linguistics*, vol. 17, no. 1, pp. 1–19, 1991.

[11] G. L. Murphy, *The big book of concepts*. MIT press, 2002.

[12] 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*, ser. SIGMOD '12. New York, NY, USA: ACM, 2012, pp. 481–492. [Online]. Available: http://doi.acm.org/10.1145/2213836.2213891

[13] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou, "Short text understanding through lexical-semantic analysis."