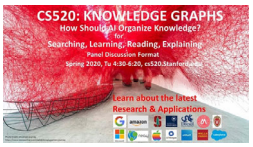

CS520

Knowledge Graphs

What should AI Know ?

How to Create a Knowledge Graph from Text?

1. Introduction

Textual sources such as business and financial news, SEC filings, and websites such as Wall Street Journal, contain information that is of great value for numerous business tasks such as market research, business intelligence, etc. We can use Natural language processing (NLP) to process textual sources, and create knowledge graphs which can support a variety of analytics. NLP, however, is a specialized and sophisticated technology, and our goal here is not to provide a comprehensive and detailed coverage of it. Our goal here is to introduce some basic concepts of NLP that can be useful to those whose primary interest and goal is to create a knowledge graph. Several vendors have created businesses around processing natural language text and delivering structured data to others for consumption.

There are three NLP tasks that are directly relevant to knowledge graph construction: entity extraction, relation extraction, and entity resolution. Entity extraction is the task of identifying key entities of interest (e.g., Organizations, People, Places, etc.) from the text. These entities typically constitute the nodes in a knowledge graph. The relation extraction is the task in which given two entities of interest, and some text, we extract the relations between them (e.g., net sales, management team information, etc.) from the text. Sometimes, the relation extraction is also used to extract properties of a given entity. The extracted relations and properties will typically become relations or node properties in our knowledge graph. Entity resolution is the task of identifying whether multiple mentions in a text refer to the same entity. For example, in a paragraph of text, "John Smith", "He", and "Her father", may all refer to the same entity.

In this chapter, we will give an overview of the techniques used for entity and relation extraction. We omit the entity resolution from our discussion here because it is an advanced topic for the scope of the present volume. Most techniques for entity and relation extraction, that are popular these days, rely on adapting a pre-trained language model for the task at hand. For our purpose, we treat the pre-trained language model and the machine learning techniques as black boxes as both are now available for use as off-the-shelf commodities. This progress in NLP and machine learning allows the knowledge graph creators to focus on the end-product, and on providing suitable training and evaluation data that is required for the adaptation of the language models. We will begin with an overview of the language models, and then describe the entity and relation extraction tasks in greater detail.

2. Overview of Language Models

Language modeling is the task of predicting what word comes next in a text. For example, given a sentence fragment: "students opened their", a language model will predict the next word that can complete this sentence. In this case, the next words might be book, exam, laptop, etc. More formally, given a set of words x_1, ... , x_{n-1}, a language model predicts the probability $P(x_n \mid x_1, \ldots , x_{n-1})$, where x_n is any word in the vocabulary. Language models are used extensively in auto completing search requests on the web, in auto correcting words in word processing, etc.

Modern language models are created by training a deep learning model, such as a Recurring Neural Network, on a large corpus of text. Numerous variations of pre-trained language models are available as open source products that can be adapted for the purpose of the specific task at hand. As we discuss the techniques for entity and relation extraction, we will also describe how the

language models can be adapted for these tasks.

3. Entity Extraction

We will begin by considering a concrete example of entity extraction, and then give an overview of different approaches to entity extraction, and conclude the section by discussing some challenges in performing well at this task.

3.1 An Example of Entity Extraction

Let us consider the following piece of text from a news story:

> Cecilia Love, 52, a retired police investigator who lives in Massachusetts, said she paid around $370 a ticket with tax for nonstop United Airlines flights to Sacramento from Boston for her niece's high school graduation in June, 2020.

We will consider the *named* entities in the above text. A named named entity is anything that can be referred to with a proper name: a person, a location, an organization, etc. The definition of a named entity is commonly extended to include things that are not entities per se, including dates, times, and other kinds of expressions involving time, and even numerical expressions, for example, prices. Here is the above text with the named entities marked:

> [$_{\text{PER}}$ Cecilia Love], 52, a retired police investigator who lives in [$_{\text{LOC}}$ New Jersey], said she paid around [$_{\text{MONEY}}$ $370] a ticket with tax for nonstop [$_{\text{ORG}}$ United Airlines] flight to [$_{\text{LOC}}$ Sacramento] from [$_{\text{LOC}}$ Boston] for her niece's high school graduation in [$_{\text{TIME}}$ June, 2020].

The paragraph contains seven named entities, one of which is a person (indicated by PER), three are locations (indicated by LOC), one is money (indicated by MONEY), one is an organization (indicated by ORG), and one is a time (indicated by TIME). Depending on the domain of application, we may introduce more or less named entity types. For example, in the task of identifying key terms in a text, there is only one entity type that captures a key term.

Entity extraction task is useful in many different applications. For example, in question answering, it may help pull out answers from a retrieved passage of text. In a word processing application, it could help automatically connect entities appearing in the text with additional information (e.g., definition, facts, etc.) about those entities.

3.2 Approaches to Entity Extraction

First and foremost, we can view entity extraction as a labeling problem. We associate a label with each word, and the task becomes to predict the label. We can perform entity extraction by three broad approaches: sequence labeling, deep learning models, and rule-based approaches. We will briefly introduce each of these approaches.

To facilitate the labeling, we introduce a labeling scheme that is known as BIOES in which the meaning of different tags is as follows: B stands for the beginning of an entity, I stands for the interior of an entity, O stands for a word that is not part of an entity, E stands for the end of an entity, and S stands for a single word entity. As an example, the words in the text snippet shown above will be labeled as shown below.

Cecilia	B	Love	E	,	O	52	O	,	O
a	O	retired	O	police	O	investigator	O	who	O
lives	O	in	O	Massachusetts	S	,	O	said	O
she	O	paid	O	around	O	$370	S	a	O

ticket	O	with	O	tax	O	for	O	nonstop	O
United	B	Airlines	E	flights	O	to	O	Sacramento	S
from	O	Boston	S	for	O	her	O	niece's	O
high	O	school	O	graduation	O	in	O	June	B
,	I	2020	E						

In the sequence labeling approach, we train one of the machine learning algorithm (for example, Conditional Random Fields), using features such as: part of speech, presence of the word in a list of standard words, word embeddings, word base form, whether word contains a prefix or suffix, whether it is in all caps, etc. Significant effort is needed in feature engineering, because the performance of a particular choice of features and the machine learning algorithm can vary by the domain.

In a deep learning approach, there is no feature engineering, and we simply input word embeddings to a language model. Instead of predicting the next word, the language model now predicts one of the five tags (B, I, O, E, S) tags that are required for entity recognition. To adapt the language model to this new task, we first pre-train it using the corpus for that domain, and then train it for the task at hand. In the task-specific training of the language model, we provide the training by adding a distinguished token [CLS] that denotes the beginning of an entity, and a second distinguished token [SEP] that denotes the end of an entity. This training allows the model to predict these distinguished tags in response to a text input. Such predictions are enough for us to produce one of the five required tags for each word.

Finally, in a rule-based approach, one specifies labeling rules in a formal query language. The rules can include regular expressions, references to dictionaries, semantic constraints, and may also invoke automated extractors and reference table structures. The rules may also invoke machine learning modules for specific tasks. Rule application can be sequenced in a way that we first use high precision rules, followed by lookup in standard name list, followed by language-based heuristics, and when all else fails, resort to probabilistic machine learning techniques.

3.3 Challenges in Entity Extraction

Even though for specific tasks, entity extractors may exhibit precision and recall above 90%, but obtaining good performance across all the domains can be challenging. In this section, we consider a few challenges that are faced during entity extraction.

While labeling entities with their classes, there are numerous cases of ambiguity. For example, given the entity *Louis Vuitton*, it can refer to either a person, or an organization, or a commercial product. Resolving such ambiguities is not possible unless considerable context is taken into account.

For using a machine learning model, we require tremendous amount of data. In practice, either the data are not available, or they are largely incomplete. When we train our model using incomplete data, it seriously affects the performance.

One variation of the entity extraction task is to identify key phrases in the text. As the key phrases are not limited to a few classes, the task of identifying the specific class corresponding to a key phrase can become even more challenging. Sometimes, the key phrases are overly complex (e.g., duplication of a cell by fission), and sometimes, too general (e.g., Attach) making it very challenging to apply a uniform technique across the board.

Entities can appear in many different surface forms, for example, synonyms, acronyms, plurals, and morphological variations. In general, entity extraction should be aware of the lexical knowledge which usually does not exist when entering a new domain. As a result, for improving the performance of entity extraction, lexicon extraction becomes an essential related task.

4. Relation Extraction

We will begin by considering several concrete examples of relation extraction, then give an overview of different approaches to relation extraction, and conclude the section by discussing some challenges in performing well at this task.

4.1 Examples of Relation Extraction

Considering the text snippet from the previous section, we can extract relations such as Cecilia Love *lives in* Massachusetts, United Airlines *flies from* Boston, and United Airlines *flies to* Sacramento, etc. In a typical relation extraction task, entities have been previously identified, and in this sense, it extends the entity extraction task. The relations to be extracted are also specified ahead of time.

A popular instance of relation extraction task is to extract information from Wikipedia Infoboxes. The obvious application of this task is to improve the search results over the internet. Wikipedia infoboxes define relationships such as *preceded by*, *succeeded by*, *children*, *spouse*, etc. Achieving a high accuracy on this task can be challenging because of numerous corner cases. For example, Larry King has been married multiple times, and therefore, the extractor must be able to take that into account the time duration for which the marriage existed.

There also exist domain-specific relationships. For example, the Unified Medical Language Systems supports relationships such as *causes*, *treats*, *disrupts*, etc. Apart from standard relationships such as *subclass-of*, and *has_part*, the relationships to be extracted are domain-specific and usually require some up front design work. There are some approaches to relation extraction that do not require choosing relationships ahead of time, but the usefulness of such approaches in practice has been found to be limited.

4.2 Approaches to Relation Extraction

There are three broad approaches to relation extraction: syntactic patterns, various forms of supervised machine learning, and unsupervised machine learning. As noted in the previous section, the unsupervised machine learning has limited use in practice. Therefore, we will primarily consider the use of syntactic patterns and supervised machine learning for relation extraction.

A classical approach to extract relations relies on syntactic patterns known as Hearst Patterns. For example, consider the following sentence.:

> The bow lute, such as the Bambara ndang, is plucked and has an individual curved neck for each string.

Even though we may have never heard of Bambara ndang, we can still infer that it is a kind of bow lute. More generally, we can identify syntactic patterns, that are strong indicators of the *subclass of* relationship. The following five syntactic patterns have been around for quite some time, and have been found extremely effective in practice.

Pattern Name	Example
such as	... works by authors *such as* Herric, Goldsmith, and Shakespear ...
or other	Bruises, wounds, broken bones, *or other* injuries ...
and other	... temples, treasuries, *and other* Civic Buildings, ...
including	All common law countries *including* Canada and England ...
especially	Most European countries *especially* France, England, and Spain, ...

We can discover new syntactic patterns for any relationship of choice as follows. We first collect

examples for which the relationship is known to hold true. We then look for sentences where the relationship holds true. By identifying commonalities across such sentences, we can define a new pattern. We can then test such patterns against the corpus. One such algorithm is known as Dual Iterative Pattern Relation Expansion (DIPRE). We illustrate its working on the problem of extracting the (author, title) pairs from a corpus. We begin with a small set set of known (author, title) pairs, and we find all their occurrences in the corpus, and from those we generate more patterns. The algorithm continues recursively by using the new patterns to discover more books, and from their discovering new patterns. As a concrete example, given the seed pair of (William Shakespear, The Comedy of Errors), and the following sentences,

- The Comedy of Errors, by William Shakespeare, was ...
- The Comedy of Errors, by William Shakespeare, is ...
- The Comedy of Errors, one of William Shakespeare's earliest attempts ...
- The Comedy of Errors, one of William Shakespeare's most ...

we can derive the following patterns:

- ?x , by ?y,
- ?x , one of ?y‘s

Using the newly derived patterns, the extraction process continues recursively.

The supervised approaches to relation extraction require extensive training data. Whenever such training data is available, many of the off-the-shelf learning algorithms could be trained. But, in the absence of adequate data, weak supervision approaches are becoming popular to arrive at the required data. The basic idea in weak supervision is to write several approximate labeling functions that can generate the training data automatically. These weak labels are then combined using a probabilistic function.

As an example of a weak labeling function, consider the *has part* relation. For this relation, it has not been possible to develop the syntactic patterns of the sort suggested above. A possible weak labeling function for this relation is to first produce a dependency parse of the sentence, and then look for two nodes in the parse that have a path of length one that contains the verbs has or have. For taxonomic relationships, an additional weak labeling function is that if two entities end with the same base word but one has an additional modifier in front of it, this suggests a taxonomic relation (e.g., eukaryotic cell SUBCLASS cell).

To adapt a language model to the task of relation extraction, we change the input representation of a sentence so that each of the individual terms are explicity marked. For example, we input a sentence such as ["All", "[TERM1-START]", "cells", "[TERM1-END]", "have", "a", "[TERM2-START]", "cell" "membrane", "[TERM2-END]", "."], and expect the model to output the probability distribution over different relationships that might exist between the two terms. The predicted relationship depends on the input training data.

4.3 Challenges in Relation Extraction

The primary challenge in relation extraction is to obtain the necessary training data. The weak supervision approach is quite promising because the training data need not be perfect. One can resort to sources such as Wikidata and Wordnet as a source of data for defining weak labeling functions. Developing new approaches for weak labeling functions is a topic of current and ongoing research.

We also need a good workflow to validate the results of the extractors. Quite often the validation can be done through crowdsourcing. One can also prioritize the validation of those extracted relations where the confidence is low. Developing such active learning loops is another important and current research challenge.

5. Summary

In this chapter, we considered the problem of creating a knowledge graph by processing text. We addressed two fundamental problems of entity extraction and relation extraction. For both of these tasks, the early work focused on defining manual and syntactic rules for extraction. More recent popular approaches rely on adapting pre-trained language models, and customizing them to the specific corpus at hand.

For both entity and relation extraction, the most prevalent current approach is to adapt a language models that has been previously created using deep learning. Existing approaches that are syntactic or rule-based play an important role in bootstrapping the training data required for the deep learning models. Validating the output of extraction remains an ongoing challenge.

The problem of entity linking or entity resolution is an equally important problem for creating knowledge graphs, but we have chosen to omit it from the discussion here for two reasons. First, we believe that the challenge of getting good performance on entity and relation extraction, by itself, is a significant hurdle. Second, a prerequisite to a good performance on entity linking is the availability of a good lexicon. For these reasons, entity resolution is an advanced technique, and it may or may not be the primary bottleneck in solving the business problem for which we are creating the knowledge graph.

Exercises

Exercise 5.1. Using the concept of a language model on the following sentence corpus, answer the questions below:

- I love running.
- Good health can be achieved by those who love running.
- I love good health.
- I love those who love running.

(*a*) What is P(health|good)?
(*b*) What is P(running|love)?
(*c*) What is P(love|I)?
(*d*) What is (good|love)?
(*e*) What is P(love|running)?

Exercise 5.2. An important feature used for entity extraction is **Word shape**: it represents the abstract letter pattern of the word by mapping lower-case letters to ‘x’, upper-case to ‘X’, numbers to ’d’, and retaining punctuation. Thus for example C.I.A. would map to X.X.X. and IRS-1040 would map to XXX-dddd. In a shorter-version of word shape, consecutive character types are removed. For example, C.I.A. would still map to X.X.X, but IRS-1040 would map to X-d. With these definitions, address the following questions.

(*a*) What is the shape of the word: Googenheim?
(*b*) What is the short-shape of the word: Googenheim?
(*c*) What is the regular expression for the shape of the word Googenheim?
(*d*) What is the regular expression for the short-shape of the word Googenheim?
(*e*) Is it true that the short-shape is always strictly smaller than the regular shape of a word?

Exercise 5.3. Which of the following may not be a good feature for learning entity extraction?

(*a*) word shape
(*b*) part of speech
(*c*) presence in Gazeteer

(*d*) presence in Wikipedia

(*e*) number of characters

Exercise 5.4. Given the following sentence corpus, and the seed (Sacramento,California), what patterns will be extracted by the DIPRE algorithm?

- The bill was signed in Sacramento, California.
- Sacramento is the capital of California.
- Sacramento is the capital of California, and its sixth largest city.
- California's Sacramento is home to the state legislature, but not the state supreme court.
- California Governor Jerry Brown signed the bill in Sacramento.

(*a*) in ?x, ?y

(*b*) ?x is the capitol of ?y

(*c*) ?y's ?x

(*d*) ?x's ?y

(*e*) ?y Governor * in ?x

Exercise 5.5. Which of the following would be a candidate for a weak labeling function to extract the parthood relationship between entities, i.e., an entity A has part entity B.

(*a*) ?xs have ?y

(*b*) ?x includes ?y

(*c*) ?x contains ?y

(*d*) ?y surrounds ?x

(*e*) ?x causes ?y