Constructing Knowledge Graph from Unstructured Text

Kundan Kumar
Siddhant Manocha

MOTIVATION

How to jointly acquire knowledge from all these sources?

Text
Video
Images
Speech/sounds
Artificial worlds?
MOTIVATION
MOTIVATION

“Where is New York City?”

Image Source: KDD 2014 Tutorial on Constructing and Mining Web-scale Knowledge Graphs, New York
PROBLEM STATEMENT

Large Amount Of Unstructured Information out there on the web!

Who is the total area of IIT Kanpur?

What is the capital of India?

Who is the director of IIT Kanpur?
KNOWLEDGE GRAPH

KNOWLEDGE GRAPH

textual abstract: summary for human

structured knowledge extraction: summary for machine

QUESTION ANSWERING

QUESTION UNDERSTANDING
1) Detection of Question Type, Focus Word and Lexical Answer Type
2) Parsing Based and Machine Learning Approaches

CONVERTING QUESTIONS TO STRUCTURED QUERIES
1) Converting natural language queries to structured database queries

QUERYING THE KNOWLEDGE BASE
1) Querying the knowledge base and retrieving the results

Who is the author of Julius Caesar?
<Who,Person> <Author,Writer> <Julius Caesar, Book>

Select author from library_db where book="Julius Caesar"

Book: Julius Caesar
Author: William Shakespeare
EXISTING KNOWLEDGE BASES

Knowledge graphs

Freebase

yago

DBpedia

Facebook’s Entity Graph

Microsoft’s Satori

Google’s Knowledge Graph

OpenIE (Reverb, OLLIE)

Image Source: KDD 2014 Tutorial on Constructing and Mining Web-scale Knowledge Graphs, New York
EXISTING KNOWLEDGE BASES

Supervised Models:
- Learn classifiers from +/- examples, typical features: context words + POS, dependency path between entities, named entity tags
- Require large number of tagged training examples
- Cannot be generalized

Semi-Supervised Models:
- Bootstrap Algorithms: Use seed examples to learn initial set of relations
- Generate +ve/-ve examples to learn a classifier
- Learn more relations using this classifier

Distant Supervision:
- Existing knowledge base + unlabeled text generate examples
- Learn models using this set of relations
OUR APPROACH

Bootstrapping Relations using Distributed Word Vector Embedding

1) Word that occur in similar context lie close together in the word embedding space.

2) Word Vectors is semantically consistent and capture many linguistic properties (like 'capital city', 'native language', 'plural relations')

3) Obtain word vectors from unstructured text (using Google word2vec, Glove, etc)

4) Exploit the properties of the manifold to obtain binary relations between entities
ALGORITHM

STEP 1: BEGIN WITH SEED EXAMPLES
ex: India, capital, Delhi
Bangladesh, capital, Dhaka

STEP 2: EXPANDING THE PRIMARY CONCEPT USING THE SEED EXAMPLES
ex: search around the space of India, Bangladesh to learn Pakistan, Maldives, Nepal, Vietnam, etc

STEP 3: LEARN THE SEMANTIC RELATIONS
Learn the relation Pakistan-Islamabad, Sri Lanka-Colombo, etc from the seed relations

STEP 4: SCORE THE LEARNED RELATIONS
Score the learned relation using the left, middle, and the right context of the words where the words occur

STEP 5: EXPAND SEED EXAMPLES BY LEARNED TRIPLETS ABOVE A THRESHOLD
SIMILARITY METRIC

\[ \text{Left context} \quad e_1 \quad \text{Middle context} \quad e_2 \quad \text{Right context} \]

\[ K(\ldots) \quad + \quad K(\ldots) \quad + \quad K(\ldots) \]

\[ \text{Left context}^* \quad e_1^* \quad \text{Middle context}^* \quad e_2^* \quad \text{Right context}^* \]

Labeled +ve or -ve example

Similarity

Test example \[ e_2^* \]
KERNEL BASED APPROACHES

1. Match attributes of parent nodes
2. If parent nodes match, add 1 to similarity score else return score of 0
3. Compare child-subsequences and continue recursively
DEPENDENCY KERNELS

1. 'his actions in Brcko’, and
2. 'his arrival in Beijing’.

1. Actual Sentences
2. Dependency Graph

1. \( x = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7] \), where \( x_1 = \{ \text{his}, \text{PRP}, \text{PERSON} \} \), \( x_2 = \{ \rightarrow \} \), \( x_3 = \{ \text{actions}, \text{NNS}, \text{Noun} \} \), \( x_4 = \{ \leftarrow \} \), \( x_5 = \{ \text{in}, \text{IN} \} \), \( x_6 = \{ \leftarrow \} \), \( x_7 = \{ \text{Brcko}, \text{NNP}, \text{Noun}, \text{LOCATION} \} \)

2. \( y = [y_1 \ y_2 \ y_3 \ y_4 \ y_5 \ y_6 \ y_7] \), where \( y_1 = \{ \text{his}, \text{PRP}, \text{PERSON} \} \), \( y_2 = \{ \rightarrow \} \), \( y_3 = \{ \text{arrival}, \text{NN}, \text{Noun} \} \), \( y_4 = \{ \leftarrow \} \), \( y_5 = \{ \text{in}, \text{IN} \} \), \( y_6 = \{ \leftarrow \} \), \( y_7 = \{ \text{Beijing}, \text{NNP}, \text{Noun}, \text{LOCATION} \} \)

Kernel:

\[ K(x,y) = 3 \times 1 \times 1 \times 1 \times 2 \times 1 \times 3 = 18 \]

3. Kernel Computation

Image Source: A Shortest Path Dependency Kernel for Relation Extraction, Mooney, et al
PRELIMINARY RESULTS

Word Vector Embedding: Wikipedia Corpus
PRELIMINARY RESULTS
(wikipedia corpus)

Seed Examples for capital relationship

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>India</td>
<td>Delhi</td>
</tr>
<tr>
<td>Bangladesh</td>
<td>Dhaka</td>
</tr>
</tbody>
</table>

Positive relations learnt

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nepal</td>
<td>Kathmandu</td>
</tr>
<tr>
<td>Afghanistan</td>
<td>Kabul</td>
</tr>
<tr>
<td>Thailand</td>
<td>Bangkok</td>
</tr>
<tr>
<td>Russia</td>
<td>Moscow</td>
</tr>
</tbody>
</table>

Negative Relations learnt

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bhutan</td>
<td>Sikkim</td>
</tr>
<tr>
<td>Algeria</td>
<td>Tunisia</td>
</tr>
<tr>
<td>Burma</td>
<td>Jalpaiguri</td>
</tr>
<tr>
<td>Kuwait</td>
<td>Cairo</td>
</tr>
</tbody>
</table>
PRELIMINARY RESULTS (google news corpus)

Seed Examples

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>India</td>
<td>Delhi</td>
</tr>
<tr>
<td>Bangladesh</td>
<td>Dhaka</td>
</tr>
</tbody>
</table>

Positive Relations Learned

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nepal</td>
<td>Kathmandu</td>
</tr>
<tr>
<td>Pakistan</td>
<td>Islamabad</td>
</tr>
</tbody>
</table>

Negative Relations Learned

<table>
<thead>
<tr>
<th>Country</th>
<th>Capital</th>
</tr>
</thead>
<tbody>
<tr>
<td>Srilanka</td>
<td>Tamil</td>
</tr>
<tr>
<td>Bhutan</td>
<td>Sikkim</td>
</tr>
<tr>
<td>Burma</td>
<td>Jalpaiguri</td>
</tr>
<tr>
<td>LTTE</td>
<td>tamil</td>
</tr>
</tbody>
</table>
References


Questions!
CBOB MODEL

- input vector represented as 1-of-V encoding
- Linear sum of input vectors are projected onto the projection layer
- Hierarchical Softmax layer is used to ensure that the weights in the output layer are between 0<=p<=1
- Weights learnt using back-propagation
- The projection matrix from the projection layer to the hidden layer give the word vector embeddings

Image Source: Linguistic Regularities in Continuous Space Word Representations, Mikolov et al 2013
### WORD VECTOR MODEL

<table>
<thead>
<tr>
<th>FRANCE</th>
<th>JESUS</th>
<th>XBOX</th>
<th>REDDISH</th>
<th>SCRATCHED</th>
<th>MEGABITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUSTRIA</td>
<td>GOD</td>
<td>AMIGA</td>
<td>GREENISH</td>
<td>NAILED</td>
<td>OCTETS</td>
</tr>
<tr>
<td>BELGIUM</td>
<td>SATI</td>
<td>PLAYSTATION</td>
<td>BLUISH</td>
<td>SMASHED</td>
<td>MB/S</td>
</tr>
<tr>
<td>GERMANY</td>
<td>CHRIST</td>
<td>MSX</td>
<td>PINKISH</td>
<td>PUNCHED</td>
<td>BIT/S</td>
</tr>
<tr>
<td>ITALY</td>
<td>SATAN</td>
<td>IPOD</td>
<td>PURPLISH</td>
<td>POPPED</td>
<td>BAUD</td>
</tr>
<tr>
<td>GREECE</td>
<td>KALI</td>
<td>SEGA</td>
<td>BROWNISH</td>
<td>CRIMPED</td>
<td>CARATS</td>
</tr>
<tr>
<td>SWEDEN</td>
<td>INDRA</td>
<td>PsNUMBER</td>
<td>GREYISH</td>
<td>SCRAPED</td>
<td>KBIT/S</td>
</tr>
<tr>
<td>NORWAY</td>
<td>VISHNU</td>
<td>HD</td>
<td>GRAYISH</td>
<td>SCREWED</td>
<td>MEGAHERTZ</td>
</tr>
<tr>
<td>EUROPE</td>
<td>ANANDA</td>
<td>DREAMCAST</td>
<td>WHITISH</td>
<td>SECTIONED</td>
<td>MEGAPIXELS</td>
</tr>
<tr>
<td>HUNGARY</td>
<td>PARVATI</td>
<td>GEFORCE</td>
<td>SILVERY</td>
<td>SLASHED</td>
<td>GBIT/S</td>
</tr>
<tr>
<td>SWITZERLAND</td>
<td>GRACE</td>
<td>CAPCOM</td>
<td>YELLOWISH</td>
<td>RIPPED</td>
<td>AMPERES</td>
</tr>
</tbody>
</table>

What words have embeddings closest to a given word? From Collobert et al. (2011)
**WORD VECTOR MODEL**

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>France - Paris</td>
<td>Italy: Rome</td>
<td>Japan: Tokyo</td>
<td>Florida: Tallahassee</td>
</tr>
<tr>
<td>big - bigger</td>
<td>small: larger</td>
<td>cold: colder</td>
<td>quick: quicker</td>
</tr>
<tr>
<td>Miami - Florida</td>
<td>Baltimore: Maryland</td>
<td>Dallas: Texas</td>
<td>Kona: Hawaii</td>
</tr>
<tr>
<td>Einstein - scientist</td>
<td>Messi: midfielder</td>
<td>Mozart: violinist</td>
<td>Picasso: painter</td>
</tr>
<tr>
<td>Sarkozy - France</td>
<td>Berlusconi: Italy</td>
<td>Merkel: Germany</td>
<td>Koizumi: Japan</td>
</tr>
<tr>
<td>copper - Cu</td>
<td>zinc: Zn</td>
<td>gold: Au</td>
<td>uranium: plutonium</td>
</tr>
<tr>
<td>Berlusconi - Silvio</td>
<td>Sarkozy: Nicolas</td>
<td>Putin: Medvedev</td>
<td>Obama: Barack</td>
</tr>
<tr>
<td>Microsoft - Windows</td>
<td>Google: Android</td>
<td>IBM: Linux</td>
<td>Apple: iPhone</td>
</tr>
<tr>
<td>Microsoft - Ballmer</td>
<td>Google: Yahoo</td>
<td>IBM: McNealy</td>
<td>Apple: Jobs</td>
</tr>
<tr>
<td>Japan - sushi</td>
<td>Germany: bratwurst</td>
<td>France: tapas</td>
<td>USA: pizza</td>
</tr>
</tbody>
</table>

Relationship pairs in a word embedding. From Mikolov *et al.* (2013b).
Figure 1: The shallow parse representation of the sentence “John Smith is the chief scientist of the Hardcom Corporation”. The types “PNP”, “Det”, “Adj”, and “Prep” denote “Personal Noun Phrase”, “Determiner”, “Adjective”, and “Preposition”, respectively.