Learning to Populate Knowledge Bases

Tom Kwiatkowski
tomkwiat@gmail.com
November 27, 2018
Outline for section 1

Graph Structured Knowledge Representations

Learning to Populate a Knowledge Base

Open Information Extraction

Universal Schema
Over the years there have been many attempts to build explicit representations of knowledge, usually via some type of *graph structure*.

- General Problem Solver - 1959
- Cyc - 1990
- Freebase - 2008
- and many others.

These have all run into problems of scalability, since they rely on human annotators.

The field of information extraction aims to populate knowledge bases automatically.
Given the size and coverage of the web, why not just use text to represent knowledge, and retrieve it for interpretation when needed?
Why Not Text as a Knowledge Representation?

Given the size and coverage of the web, why not just use text to represent knowledge, and retrieve it for interpretation when needed?

- Web text is extremely redundant, while knowledge bases attempt to express each fact only once.
- Knowledge bases support interpretable reasoning, since we can follow a chain of assertions. This is not true of current work in reading comprehension etc.
- We can do inference on knowledge bases to learn rules
  - \( \text{child}(x, y) \land \text{child}(y, z) \rightarrow \text{grandchild}(x, z) \).
Why Not Text as a Knowledge Representation?

Given the size and coverage of the web, why not just use text to represent knowledge, and retrieve it for interpretation when needed?

- Web text is extremely redundant, while knowledge bases attempt to express each fact only once.
- Knowledge bases support interpretable reasoning, since we can follow a chain of assertions. This is not true of current work in reading comprehension etc.
- We can do inference on knowledge bases to learn rules
  \[ \text{child}(x, y) \land \text{child}(y, z) \rightarrow \text{grandchild}(x, z). \]

Note

Some people believe that we don’t need any static representation of knowledge other than text and the parameters of a neural network that can interpret it in the context of a given task. This is still an open question.
Outline for section 2

Graph Structured Knowledge Representations

Learning to Populate a Knowledge Base

Open Information Extraction

Universal Schema
Let’s work through the information extraction system introduced by [Hoffmann et al., 2011]. This assumes as input

- a set of entities $E$
- a set of relations $R$
- a set of ground facts $\Delta$, each containing two entities and one relation
- a training corpus of sentences $\Sigma$ in which entity mentions have been resolved to entities in $E$

The goal is to learn an extractor that can extract new facts from text. The extractor is trained through \textit{distant supervision}, since we don’t actually know which sentences in $\Sigma$ express the facts in $\Delta$. 
With distant supervision, we don’t know which mentions of an entity pair \((e_1, e_2)\) express a particular fact. So we can treat fact mentions as \textit{latent}.

Each mention of the entity pair \((e_1, e_2)\) has a latent variable \(Z_i\) that ranges over the relations \(r \in R\) that could be expressed.

There is a global variable \(Y\) for each \(r \in R\) and entity pair \((e_1, e_2)\) that indicates whether the entire corpus expresses the fact \(r(e_1, e_2)\).
Explicitly:

\[ P(Y = yA; Z = z|x; \theta) := \frac{1}{Z_x} \prod_{r \in R} \Phi^{\text{join}}(y^r, z) \prod_i \Phi^{\text{extract}}(z_i, x_i) \]

where \( \Phi^{\text{extract}} \) operates on individual mentions of each entity pair \( x_i \)

\[ \Phi^{\text{extract}}(z_i, x_i) = \exp(\theta \cdot \phi(z_i, x_i)) \]

and \( \Phi^{\text{join}} \) is simply a deterministic OR operation

\[ \Phi^{\text{join}} = \begin{cases} 1, & \text{if } y^r = \text{true and } \exists i : z_i = r \\ 0, & \text{otherwise.} \end{cases} \]

This formulation decouples the global supervision that comes from an existing knowledge base from the local model of relation mentions.
Define the training set to be \( \{(x_i, y_i) | i = 1 \ldots N\} \) where:

- each \( i \) indicates a particular entity pair \((e_j, e_k)\)
- \( x_i \) contains all sentences mentioning this pair
- \( y_i \) contains all relations that link the pair.

The training objective is then

\[
\hat{\theta} = \arg \max_{\theta} \prod_{i} \sum_{z} p(y_i, z|x_i, \theta)
\]

which is hard to calculate. [Hoffmann et al., 2011] simplifies learning by using online learning with local Viterbi approximations.

Inference is a lot easier, since \( \Phi^{\text{join}} \) is deterministic. So we can perform local extractions, and then use dynamic programming to find the global assignment that best describes the data.
Decoupling the corpus wide supervision from the sentence-local predictions massively increases precision for both the corpus level predictions, and the per-sentence predictions.
Outline for section 3

Graph Structured Knowledge Representations

Learning to Populate a Knowledge Base

Open Information Extraction

Universal Schema
Open Information Extraction

We just went through an example of information extraction for a fixed ontology in which all relations and entities are predefined. This approach will probably never scale to cover the massive set of relations that we can express using language.

\[(\text{Faust, made a deal with, the devil})\]

— [Fader et al., 2011]

*Open information extraction* extracts assertions without any dependence on an entity or relation vocabulary.
Again, we will walk through an illustrative paper [Fader et al., 2011], which extracts (entity, relation, entity) triples as follows

1. Find relations that match a regex over POS tags, and pass a frequency threshold.
2. Find surrounding entities to fill the subject and object slots.

Once relations have been identified, entities are simply noun phrases to the left (subject) and right (object).

Relation identification is similarly simple, but surprisingly effective.
Relations are identified with the following regex over POS tags.

\[
[ V \mid VP \mid VW^*P ]^+
\]

V: verb particle? adv?
W: (noun, adj, pron, det)
P: (prep, particle, in. marker)

which matches ‘invented’, ‘located in’, ‘has atomic weight of’, ‘wants to extend’. The authors claim that this covers 85% of relations that they found in a sample of web text.

Beyond this simple regex, each relation surface form must occur with at least \( k \) entity pairs in the web corpus used.
The lexical constraint, which requires each relation form to have been seen with multiple argument pairs, clearly aids performance in all P/R regimes.
Outline for section 4

Graph Structured Knowledge Representations

Learning to Populate a Knowledge Base

Open Information Extraction

Universal Schema
Information extraction with a fixed ontology maps distinct surface forms onto a single canonical representation.

\[
\begin{align*}
& x \text{ married } y \\
& x \text{ is married to } y \\
& x \text{ wed } y \\
& x \text{ tied the knot with}
\end{align*}
\rightarrow \text{spouse}_\text{of}(x, y)
\]

But fixed ontologies are very limiting. Open information extraction allows an unbounded number of relations, but these are very inefficient.

The *universal schema* [Riedel et al., 2013] attempts to get the best of both worlds by mapping free text relations into the same space as relations from a predefined ontology.
In collaborative filtering, such as work in probabilistic analogies, we can draw from a large body of work. The probability that a customer will estimate this using a method involves estimating weights and vectors. In section 2.5, we will introduce a series of exponential family models that allow us to formalize this estimation in a joint fashion.

The universal schema incorporates these tasks in a structured relation, such as alignment and integration models. Correlations between patterns, and can be fed into RE (Yao et al., 2011). Database surface form clustering models correlations between pattern relations (and other features) to structured relations.

Notice that we can interpret the natural parameter as the logarithm of the odds ratio. This corresponds to generalized PCA (Collins et al., 2001), a model where the matrix parameters is defined as the low rank factorization $\theta = AV$.

For tuple $(r,t)$, the neighborhood model amounts to a collection of local log-linear classifiers, one for each relation $r$, such that $p(r,t) := 1 + \exp(-\theta_{r,t})$.

**Objective**

Learn $\theta_{r,t}$ such that $p(y_{r,t} = 1|\theta_{r,t}) := \frac{1}{1 + \exp(-\theta_{r,t})}$ models the probability of $r$ holding between the entities in $t$.

<table>
<thead>
<tr>
<th>Surface Patterns</th>
<th>KB Relations</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-professor-at-Y</td>
<td>employee(X,Y)</td>
</tr>
<tr>
<td>X-historian-at-Y</td>
<td>member(X,Y)</td>
</tr>
</tbody>
</table>

**Columns**

represent relation patterns $r$ that come from a KB or text.

**Rows**

represent entity pairs $t$.

**Clusters**

relate to queries on a per-relation basis, and a per-relation bias-terms. In section 4, we evaluate ranked answers.

### Figure 1: Filling up a database of universal schema.

<table>
<thead>
<tr>
<th>Ferguson Harvard</th>
<th>Oman Oxford</th>
<th>Firth Oxford</th>
<th>Gödel Princeton</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0.95</td>
<td>1</td>
<td>0.95</td>
</tr>
<tr>
<td>1</td>
<td>0.05</td>
<td>0.93</td>
<td>0.97</td>
</tr>
</tbody>
</table>

### Rows

- Cluster
- Rel. Extraction
- Align
- Reasoning with Universal Schema

**Align**

Surface patterns KB relations

- X-professor-at-Y
- X-historian-at-Y
- employee(X,Y)
- member(X,Y)
[Riedel et al., 2013] propose the decomposition \( \theta_{r,t} = \theta^F_{r,t} + \theta^N_{r,t} + \theta^E_{r,t} \)
Universal Schema - Model

[Riedel et al., 2013] propose the decomposition $\theta_{r,t} = \theta^F_{r,t} + \theta^N_{r,t} + \theta^E_{r,t}$

Latent Feature Model

$$\theta^F_{r,t} = a_r \cdot v_t$$
[Riedel et al., 2013] propose the decomposition $\theta_{r,t} = \theta^F_{r,t} + \theta^N_{r,t} + \theta^E_{r,t}$

**Latent Feature Model**

$$\theta^F_{r,t} = a_r \cdot v_t$$

**Neighborhood Model**

$$\theta^N_{r,t} = \sum_{(r',t) \in \mathcal{O} \setminus \{(r,t)\}} w_{r,r'}$$

models relation pair interactions.
[Riedel et al., 2013] propose the decomposition $\theta_{r,t} = \theta_{r,t}^F + \theta_{r,t}^N + \theta_{r,t}^E$

**Latent Feature Model**

$$\theta_{r,t}^F = a_r \cdot v_t$$

**Neighborhood Model**

$$\theta_{r,t}^N = \sum_{(r',t) \in O \setminus \{(r,t)\}} w_{r,r'}$$

models relation pair interactions.

**Entity Model**

$$\theta_{r,t}^E = d_{r_1} \cdot t_{t_1} + d_{r_2} \cdot t_{t_2}$$

models interaction between relation slots and individual entities $t_1$ and $t_2$. 
How do we learn from only sparse positive observations?

**Dataset of Ranked Pairs**

For each observed fact \( r, t^+ \in \mathcal{O} \), choose all \( t^- \) s.t. \( r, t^- \notin \mathcal{O} \). and maximize

\[
\sum_{r, t^+ \in \mathcal{O}} \sum_{r, t^- \notin \mathcal{O}} \log(\sigma(\theta_{r,t^+} - \theta_{r,t^-}))
\]

where

\[
\sigma(x) = \frac{1}{1 + \exp(-x)}
\]

In practice, the learning algorithm samples unseen facts rather than summing over all possible unseen facts. The ranking loss is robust even though many unseen facts will actually be true.
In the low precision, high recall regime the feature model (F) is not helped by the neighborhood model (N) or the entity model (E). But the neighborhood model does help a lot at higher precisions.
The initial formalism requires a separate column for every surface form realization of a relation, and a separate row for each distinct entity pair.

- [Singh et al., 2015] and [Verga and McCallum, 2016] reduce the reliance on separate entity pair representations
- [Toutanova et al., 2015] reduce the reliance on separate relation representations

[Das et al., 2017] use the universal schema as part of an end-to-end learned question answering system.
