Objectives for Information Extraction

Justin T. Chiu

June 27, 2019

Abstract

Many recent information extraction systems predict the relationship between an entity and value given their locations in the text. This requires observed locations of mentions, which requires annotations at the word level. Any form of annotation at the word level does not scale as the size of the text and the number of labels increases, and even more so if there is ambiguity. In order to train a probabilistic information extraction without any supervision at the level of text, we specify a model that, for each word, either chooses a triple from a knowledge base to explain or chooses to explain nothing.

1 Problem Setup

In relation extraction (RE) we extract facts from a passage of text. The goal of RE is to convert facts expressed in natural language into a form amenable to computation. RE focuses on a relational representation: facts are extracted into a form that details how values are related to entities.

We assume supervision only at the proposition level of a KB, therefore we must locate the expression of a fact in text. As sentences may contain many entities and values, and therefore possibly a large number of facts, we focus on identifying facts at the word level. In particular, we focus on identifying the location of value mentions and predicting who (the entity) and what (the relation type) the value mention is discussing.

Note: Except for Zeng et al. (2018), prior work has either assumed that the locations of entities and values are given as input features or that the locations of entities and values are observed at training time.

We are primarily concerned with the scenario where we have an overcomplete KB schema with respect to a specific passage of text. This fits many scenarios in real world applications: we may have thousands of entities of interest if our KB was pulled from an external source such as Freebase, but the particular document we wish to analyze only discusses tens of entities, only a few of which are present in our KB.

Given a text $x = [x_0, \ldots, x_I]$ we must model the facts $r = \{(e_j, t_j, v_j)\}_{j \in J}$ expressed in that text with respect to a schema that details all entities $e_j \in \mathcal{E}$, relation types $t_j \in \mathcal{T}$, and all values $v_j \in \mathcal{V}$. We assume that the schema of the KB $(\mathcal{E}, \mathcal{T}, \mathcal{V})$ is known at all times, and that the schema covers all facts of interest.

Table 1: Notation

x	$\triangleq$	A sequence of words corresponding to a sentence or document.
r	$\triangleq$	The knowledge base, an indexed set of records each consisting of entities, types, and values.
e	$\triangleq$	The list of all entities in a KB by index.
t	$\triangleq$	The list of all types in a KB by index.
v	$\triangleq$	The list of all values in a KB by index.
$f_*(x)$	$\triangleq$	A learned function whose output is of shape $T \times D$, where D is the context-dependent.
$c_t \sim \text{Bern}(f_c(x)_t)$	$\triangleq$	A word level RV that indicates whether a word is content or not. We have $c_t \in \{0,1\}$.
$a_t \sim \text{Cat}(f_a(x)_t)$	$\triangleq$	A word level RV that indicates an the index of the aligned for word x_t. We have $a_t \in J$.
$z_t \sim \text{Cat}(f_z(x)_t)$	$\triangleq$	A word level RV that indicates the canonical value word x_t is translated to, such that $z_t \in \mathcal{V}$

We refer to the set of facts r as a knowledge base (KB), and each fact r_j is referred to as a record. Each record r_j consists of an entity, type, and value triple. Let e, t, v be the projection of the entities, types, and values from the KB r in aggregate.

We wish to model the distribution $p(v \mid x, e, t)$, the distribution over values of corresponding entities and types given the text. We omit the conditioning on e, t for brevity.

2 Model

We introduce the following random variables:

1. $c_t \sim \text{Bern}(f_c(x)_t)$: A word level RV that indicates whether a word is content or not. We have $c_t \in \{0,1\}$.

2. $a_t \sim \text{Cat}(f_a(x)_t)$: A word level RV that indicates an the index of the aligned for word x_t. We have $a_t \in J$.

3. $z_t \sim \text{Cat}(f_z(x)_t)$: A word level RV that indicates the canonical value word x_t is translated to, such that $z_t \in \mathcal{V}$.

We propose the following factorization

$$p(y \mid x) = \sum_{a,c} \prod_j p(v_j \mid x) \tag{1}$$

Our information extraction model is denoted $q(r \mid y)$, a distribution over a KB r given some text y. Let f, g, h be arbitrary learned functions. Our word level model is given by the following: For each word y_t, we have

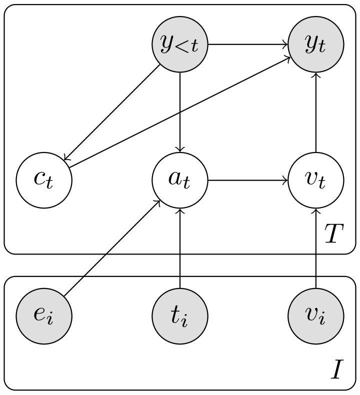

Figure 1: The generative model which produces words given a knowledge base.

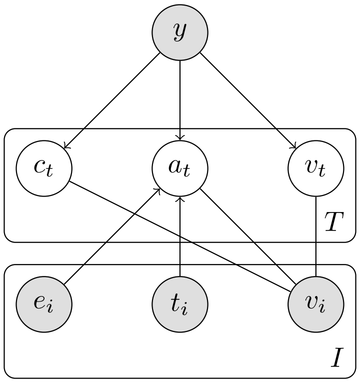

Figure 2: The inference network which predicts word-level values and alignments.

1. Value mention detector $q(c_t \mid y)$: We classify whether word y_t is a value mention. Each $c_t \sim \text{Bern}(f(y))$, where a value of 1 indicates that y_t is a value mention.

2. Translation $q(v_t \mid y)$: We translate the word y_t into a value v_t. The $v_t \sim \text{Cat}(g(y))$ is the canonical value from the KB schema associated with y_t.

3. Alignment $q(a_t \mid y)$: We align the word y_t by classifying who (the entity) and what (the relation type) are being talked about. In particular, $a_t \sim \text{Cat}(h(y))$ denotes the alignment to the record r_{a_t} given by the index $i = a_t$.

Finally, we aggregate the word level information at the sequence level:

1. Aggregation $q(r_i \mid a, v, c)$: In order to parameterize a distribution over the KB r, we must then aggregate all of the word-level information into a distribution over facts at the sequence level. We use a deterministic OR potential for each r_i (stuck here, need to read up on this).

3 Three perspectives on training

We can either train q directly on the conditional task or train it to mimic the posterior of a suitable generative model.

Axes of objectives:

1. Proposal distribution: Learned or uniform (or some prior)
2. Probabilistic interpretation: Marginal likelihood or KL
3. Probabilistic interpretation 2: Approximate posterior of a generative model or learn directly.

3.1 Marginal loss

Our first loss, the marginal likelihood, takes the form of

$$\mathcal{L}_{\text{ML}} = \sum_t \log \sum_{a_t} \sum_{c_t} q(a_t, c_t, v_{a_t})$$

where the unobserved alignments and latent copy are marginalized over. This has the interpretation of maximizing the 'softmax' of the log probabilities, where softmax is from the physics usage as a smooth approximation to the max. However, the sharpness of is limited by the choice of q. With a uniform $q(a,c)$, $\mathcal{L} = \sum_t \sum_{a_t} q(v_{a_t}) + C$, where C is the normalization term which we can assume to be constant here. The gradient of this term wrt the log probabilities is weighted by the posterior, so a high entropy q will result in a smaller gradient.

WRONG! The inference network is a mixture model consisting of $q(v \mid y)$ and $p(v)$! Derive from generative model.

3.2 KL

The second loss, a lower bound on the marginal likelihood, is the following

$$\mathcal{L}_{\text{KL}} = \sum_t \sum_{a_t} \sum_{c_t} q(a_t, c_t) \log q(v_{a_t})$$

The benefit of this loss is that we can approximate it via Monte Carlo sampling. Under a uniform $q(a,c)$, we must maximize the probabilities of all values. With a learned q, the lower bound can be made much tighter so there is less of a difference.

3.3 Approximating the posterior of a generative model

The information extraction model itself was inspired by the following generative process: For every word y_t

1. Generate the values of the KB $v \sim p(v)$ assuming the schema is fixed.

2. Choose $c_t \sim \text{Bern}(f(y_{<t}))$, which determines whether to generate word y_t from a language model or the KB.

3. If $c_t = 0$, generate $y_t \sim \text{Cat}(g(y_{<t}))$.

4. Otherwise pick an alignment to the KB $a_t \sim \text{Cat}(h(y_{<t}))$ then generate $y_t \sim \text{Cat}(f'(v_{a_t}))$.

We then train our model to approximate the posterior $p(v \mid y)$. Since this generative model does not use every record during generation, the posterior may only explain a subset of all records. This fits our hypothesis that the KB contains a superset of records expressed in text.

The loss is then given by

$$\mathcal{L}_{\text{VI}}$$

The primary benefit of this over a purely conditional approach is semi-supervised information extraction, where values are missing.

$$\begin{aligned}
\arg\max_p \sum_{y'} \sum_{x'} p^*(y', x') \log \frac{p(y', x')}{p^*(y', x')} &= \arg\max_p \sum_{y'} \sum_{x'} p^*(y', x') \log \sum_z p(y', z, x') && (2)\\
&= \arg\max_p \sum_{y'} \sum_{x'} p^*(y', x') \log \sum_z p(y', z, x') && (3)\\
&\geq && (4)
\end{aligned}$$

References

Xiangrong Zeng, Daojian Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Extracting relational facts by an end-to-end neural model with copy mechanism. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 506–514, Melbourne, Australia. Association for Computational Linguistics.