Objectives for Information Extraction

Justin T. Chiu

July 4, 2019

Abstract

Many recent information extraction systems predict the relationship between an entity and value given the positions of their mentions in the text. This requires requires words to be annotated as mentions. Human annotation at the word level does not scale as the size of the text and the number of labels increases, as annotators must read every word. Automatic methods allow annotation to scale, but may introduce noise due to incorrect annotations. In order to train a probabilistic information extraction model without mention annotations, 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 Statement

In relation extraction the goal is to extract facts from a passage of text. Systems must convert facts expressed in natural language into a form amenable to computation. Facts consist of three components: entities, relation types, and values. The challenge is to not only extract facts from text, but also justify the extractions by determining where those facts are mentioned.

A mention is a surface realization of an abstract object in text. In relation extraction we justify extractions by identifying fact mentions. As text is noisy, the realization of a fact may be difficult to locate. We focus on locating fact mentions at the word level by identifying individual words as value mentions, rather than entity or type mentions.

We propose a model that first identifies value mentions at the word level, aligns those mentions to an entity and relation type in order to obtain a fact, then aggregates word level decisions to resolve conflicts. We assume access to a set of facts, henceforth referred to as a knowledge base (KB), that is discussed in text.

The problem description is as follows: given a text $x = x_1, \ldots, x_I$ we model the facts $\Delta = \{(e_j, t_j, v_j)\}_{j=1}^{J}$ expressed in that text with respect to a schema that details all entities e_j, relation types t_j, and all values v_j. The set of facts Δ is our knowledge base (KB). Let $s = \{(e_j, t_j)\}_{j=1}^{J}$, and $v = \{v_j\}_{j=0}^{J}$.

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 all entities e and types t, we reduce the construction of r to predicting, for every $j \in 1, \ldots, J$, the value v_j corresponding to the entity e_j and type t_j. This reduction is inspired by the representation of a KB as a table: the rows of the table are given by the entities e, the columns by the types t, and the cells of the table take on the values v. In the following section we propose a model for the distribution $p(v \mid x, e, t)$.

2 Model

We define a graphical model that performs extraction with justification. The model first extracts information at the word level, then aggregates its choices for each word into an extraction at the sequence level.

The word level extraction process has three steps. For each index $i \in 1, \ldots, I$ we perform

1. Value mention identification: Given a sequence of words x, we identify whether each word is a value mention with $p(m \mid x) = \prod_i p(m_i \mid x)$. Each $m_i \in \{0, 1\}$. Not every word in a mention must be identified; it suffices to find at least one word in a value mention.

2. Alignment: Each value mention is then aligned to a record in the knowledge base with $p(a \mid x, e, t) = \prod_i p(a_i \mid x, e, t)$, We align the word x_i by choosing who (the entity) and what (the relation type) generate the possible value mention at index i. In particular, $a_i = j$ denotes the alignment to the record r_j with $a_i \in 1, \ldots, J$. We assume that each value mention aligns to a single record.

3. Translation: All value mentions are translated into a value from the KB schema with $p(z \mid x) = \prod_i p(z_i \mid x)$, with $z_i \in \mathcal{V}$.

Finally, we aggregate the word level information at the sequence level in order to give a single distribution over the record values for x.

4. Aggregation $p(v \mid z, a, m) = \prod_j p(v_j \mid z, a, m)$: Given the word level values z, alignments a, value mention decisions m, we choose the sequence level value v_j.

This gives us the following factorization of the relation extraction system:

$$\begin{aligned} p(v \mid x, e, t) &= \sum_{z,a,m} p(v, z, a, m \mid x, e, t) \\ &= \sum_{z,a,m} p(v \mid z, a, m, x, e, t) p(z, a, m \mid x, e, t) \\ &= \sum_{z,a,m} \left(\prod_j p(v_j \mid z, a, m, x, e, t) \right) \left(\prod_i p(z_i \mid x) p(a_i \mid x, e, t) p(m_i \mid x) \right) \end{aligned} \tag{1}$$

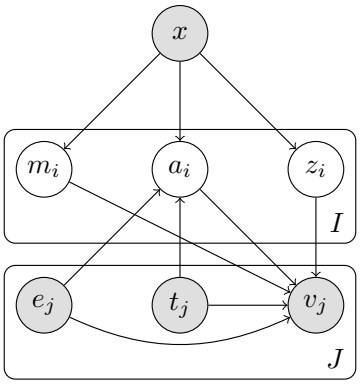

Figure 1: Our model predicts word-level values and alignments then aggregates those choices over all indices i to predict values at the KB level.

2.1 Parameterization

Our model has four steps: mention identification, mention alignment, mention translation, and aggregation. We parameterize the conditional distributions of each step below.

Let $\mathbf{h}_i \in \mathbb{R}^d$ be a contextual embedding of the word x_i, and E an embedding function that maps entities and types to vectors in $\mathbb{R}^{d'}$.

1. Identification: We use the contextual embedding to directly predict whether a word is part of a value mention.

$$p(m_i \mid x) \propto \exp(W_m \mathbf{h}_i), W_m \in \mathbb{R}^{2 \times d}$$

2. Alignment: We decompose the alignment distribution into a distribution over entities $p(\epsilon_i \mid x, e)$ and types $p(\tau_i \mid x, t)$.

$$\begin{aligned} p(a_i \mid x, e, t) &= p(\epsilon_i \mid x, e) p(\tau_i \mid x, t) \\ p(\epsilon_i \mid x) &\propto \exp(E(e_{\epsilon_i})^T W_e \mathbf{h}_i) \\ p(\tau_i \mid x) &\propto \exp(E(\tau_{a_i})^T W_t \mathbf{h}_i) \end{aligned}$$

with $W_e \in \mathbb{R}^{d' \times d}, W_t \in \mathbb{R}^{d' \times d}$.

3. Translation: We use the contextual embedding to translate a word into a value.

$$p(z_i \mid x) \propto \exp(W_z \mathbf{h}_i), W_z \in \mathbb{R}^{|\mathcal{V}| \times d}$$

4. Aggregation: If there exists an index that is a mention and is also aligned to r_j we allow it to vote on the value v_j, otherwise we ignore the text and use a prior distribution over values $p(v_j \mid e_j, t_j) \propto \exp(E(v_j)^T W_v [E(e_j), E(t_j)])$.

$$p(v_j \mid z, a, m, e, t) \propto \begin{cases} \prod \exp(\psi(v_j, z_i, a_i, m_i, e, t)), & \exists i, m_i = 1 \wedge a_i = j \\ \exp(E(v_j)^T W_v [E(e_j), E(t_j)]), & \text{otherwise} \end{cases}$$

$$\psi(v_j, z_i, a_i, m_i, e, t) = 1(v_j = z_i, a_i = j, m_i = 1)$$

3 Training and Inference

To train a latent variable model, we must marginalize over the unobserved RVs and maximize the likelihood of the observed. Ideally, we would optimize the following objective

$$\log p(v \mid x, e, t) = \log \sum_{z,a,m} p(v, z, a, m \mid x, e, t) \tag{2}$$

However, maximizing $\log p(v \mid x)$ directly is very expensive for this model as the summation over z, a, m is intractable. The summation over z, a, m has computational complexity $O((|\mathcal{V}| \cdot J \cdot 2)^I)$, which is exponential in the length of the text. Additionally, the size of the KB J may be large as well.

We therefore resort to approximate inference, specifically amortized variational inference.

3.1 Inference network

Our first approach is to introduce an inference network $q(z, a, m \mid v, x, e, t)$ and optimize the following lower bound on the marginal likelihood with respect to the parameters of both p and q:

$$\log p(v \mid x) \geq \mathbb{E}_{q(z,a,m|v,x,e,t)} \left[\log \frac{p(v, z, a, m \mid x, e, t)}{q(z, a, m \mid v, x, e, t)} \right] \tag{3}$$

We propose to parameterize $q(z, a, m \mid v, x, e, t)$ as follows. We decompose

$$\begin{aligned} q(z, a, m \mid v, x, e, t) &= q(z \mid a, v, x) q(a \mid v, x, e, t) q(m \mid v, x) \\ &= \prod_i q(z_i \mid a, v, x) q(a_i \mid v, x, e, t) q(m_i \mid v, x) \end{aligned} \tag{4}$$

The conditional distributions of our inference network are very similar to the relation extraction model, but they condition on the values v.

Let $\mathbf{h}_i \in \mathbb{R}^d$ be a contextual embedding of the word x_i. We use attention weights over records to get a weighted representation of the records of the KB for each index i:

$$\begin{aligned} \mathbf{g}_{r_j} &= [E(e_j), E(t_j), E(v_j)] \\ \alpha_j &\propto \exp(\mathbf{g}_{r_j}^T W_\alpha \mathbf{h}_i) \end{aligned}$$

The inference network is given by

1. The value mention model $q(m_i \mid v, x)$ has access to the values v from the KB, which it conditions on when detecting value mentions.

$$p(m_i \mid v, x) = W'_m \text{MLP}([\sum_j \alpha_j \cdot \mathbf{g}_{r_j}, \mathbf{h}_i]), W'_m \in \mathbb{R}^{2 \times d}$$

2. The alignment model $q(a_i \mid v, x, e, t)$ uses a contextual representation of each x_i and chooses a record. In contrast to $p(a \mid x, e, t)$, this model has access to values as well. We use the attention weights to parameterize the distribution $p(a_i \mid x) = \alpha_{a_i}$.

3. The translation model $q(z_i \mid a, v, x) = 1(z_i = v_{a_i})$ conditions on the alignments a and ensures the chosen z is consistent with the alignments.

One concern is that the model may learn to never rely on the text for extraction, setting $m_i = 0$ at every index. We can avoid this by initializing $q(z)$ to ensure that for words $x \in \mathcal{V}$ we have $q(z = x)$ is high, biasing the translation model towards transliteration at the start of training.

3.2 Approximate the posterior of a generative model

Alternatively, we may introduce a generative model of text $q(x, v)$ that inverts the relation extraction model $p(v \mid x)$ and optimize the following objective:

$$\log q(x, v) - KL[p(v, z, a, m \mid x)||q(v, z, a, m \mid x)] \tag{5}$$

which entails training the generative model of text while approximating its posterior with the information extraction system.

decompose the training of our extraction system $p(v \mid x)$ into two stages: In the first stage we train $p(z, a, m \mid x, e, t)$ to approximate the posterior of a conditional model of text given a complete KB $q(x, z, a, m \mid e, t, v)$. This has the benefit of allowing us to exert control over where value mentions are detected through our design of the text model q.

In the second stage, we have two choices: a) train $p(v \mid z, a, m, x, e, t)$ to approximate the posterior of a full generative model of text and the values of KB $q(x, v \mid e, t)$. b) train $p(v \mid z, a, m, x, e, t)$ using the following lower bound:

$$\log p(v \mid x) \geq \mathbb{E}_{p(z,a,m|x,e,t)} \left[\log p(v \mid z, a, m, x, e, t)\right] \tag{6}$$

Ideally the bound in Eqn. 6 should not be looser than the one presented in Eqn. 3, as conditioning on the observed values of a KB should not reduce the entropy of a good alignment model.

How do we split the gradient over time?

4 Evaluation

Although we have a model over the values of all records, evaluation does not include the final distribution over all record values. As we assumed that the KB contained a superset of the facts contained in a sequence of text, we are evaluate whether the model can discover the subset of facts that are expressed in the text. We therefore perform extraction by using the marginal distributions $q(z), q(a), q(c)$ to value mentions as well as entities and types, giving us facts.

References