Relation Extraction

Justin T. Chiu

July 12, 2019

Abstract

Recent relation extraction systems predict the relationship between an entity and value given the positions of their mentions in the text. This requires words to be annotated as mentions. The cost of obtaining human annotations for each word scales linearly with the size of the text. Automatic annotation methods allow the annotation process to scale sublinearly in human effort, 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, identifies important words and uses them to explain a triple from a knowledge base.

1 Problem Statement

Relation extraction aims to extract facts from a passage of text. Extraction systems 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.

The problem description is as follows, focusing on the domain of basketball summaries: Given a written summary of a basketball game $x = x_1, \ldots, x_I$ of length I, we model the aligned box score $\{(e_j, t_j, v_j)\}_{J=1}^{J}$ consisting of entities e_j, relation types t_j, and all values $v_j \in \mathcal{V}$. The set of facts is our knowledge base (KB), which contains J facts.

Let $D = \{(e_j, t_j)\}_{j=1}^{J}$ and $v = \{v_j\}_{j=1}^{J}$. The KB (D, v) can be viewed as a data table where d defines a flattened representation of the rows and columns and v gives the values of the cells. Our goal is to locate and extract facts from x.

Modeling only the KB (d, v) given the text x is not sufficient, as our goal is to locate fact mentions. In fact, we assume the KB contains many more facts than those mentioned in the text. This fits many scenarios in real world applications: we may have many entity and type pairs in our data table, but a summary may discuss only a small, salient subset of players and statistics. We therefore propose a model that first identifies words as value mentions, aligns those mentions to an entity and relation type in order to obtain a fact, and then aggregates word level decisions to resolve conflicts.

2 Model

We define a graphical model that performs extraction with justification.

2.1 Notation

We first introduce the latent variables of our model. Consider the follow example: Let our KB consist of the data table

$$D = \begin{cases} (e_1 = \text{John Doe}, t_1 = \text{Points}), \\ (e_2 = \text{John Doe}, t_2 = \text{Rebounds}), \\ (e_3 = \text{John Doe}, t_3 = \text{First name}), \\ (e_4 = \text{John Doe}, t_4 = \text{Last name}), \\ \dots \end{cases}$$

with associated values $v = \{v_1 = 19, v_2 = 12, v_3 = \text{John}, v_4 = \text{Doe}, \dots\}$ aligned to the brief summary $x = \text{John Doe scored 19 points}$.

1. $m = m_1, \dots, m_I$ where each $m_i \in \{0, 1\}$ indicates whether word x_i is part of a value mention. In our example, we have $m_1 = 1$ as $x_1 = \text{John}$ mentions the value v_3, the first name of the entity 'John Doe'.

2. $a = a_1, \dots, a_I$ where each $a_i \in \{1, \dots, J\}$ gives the index of the fact whose value word x_i mentions.

3. $z = z_1, \dots, z_I$

4. $v = v_1, \dots, z_J$

2.2 Model

The word level extraction process has three steps. For each index $i \in 1, \dots, I$ 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 $p(a \mid x, d) = \prod_i p(a_i \mid x, d)$, 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 $a_i \in 1, \dots, J$. 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 $p(v \mid z, a, m, d) = \prod_j p(v_j \mid z, a, m, d_j)$: Given the word level values z, alignments a, value mention decisions m, and data table d we choose the sequence level value v_j.

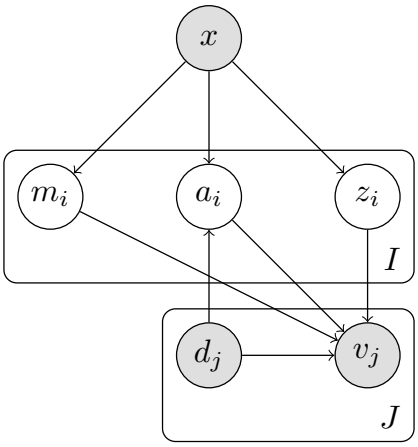

Figure 1: Our model predicts word-level values and alignments then aggregates those choices over all indices i to make a single decision for each value. Each word has the following latent variables: the mention $m_i \in \{0, 1\}$ indicates whether word x_i is a value mention, the alignment a_i gives what fact x_i aligns to, and the value z_i gives the canonical value that x_i translates to.

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

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

2.3 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 $p(\epsilon_i \mid x, d)$ and types $p(\tau_i \mid x, d)$.

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

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 $p(v_j \mid d_j) \propto \exp(E(v_j)^T W_v [E(d_j)])$.

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

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

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

However, maximizing $\log p(v \mid x, d)$ 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 $q(z, a, m \mid v, x, d)$ and optimize the following lower bound on the marginal likelihood with respect to the parameters of both p and q:

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

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

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

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:

$$\mathbf{g}_{r_j} = [E(d_j), E(v_j)]$$
$$\alpha_j \propto \exp(\mathbf{g}_{r_j}^T W_\alpha \mathbf{h}_i)$$

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 $q(a_i \mid v, x, d)$ uses the attention weights to parameterize the alignment $p(a_i \mid x) = \alpha_{a_i}$.

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.

We use the score function gradient estimator to perform gradient ascent on the objective in Equation 4 with respect to both p and q. We utilize a leave-one-out baseline for variance reduction.

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.

4 Evaluation

As we assumed that the KB contained a superset of the facts contained in a sequence of text, we are interested in evaluating whether the model can discover and locate the subset of facts that are expressed in the text.

We evaluate by determining whether the model can identify which facts in a KB are expressed in text. Given a ground truth set of facts extracted by a human annotator, we compute the precision and recall of the extractions by the model. We perform extraction by finding $\arg\max_{z_i,a_i,m_i} q(z_i), q(a_i), q(m_i)$ for each word x_i.

References