Relation Extraction

Justin T. Chiu

July 29, 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.

The problem description is as follows: Given a textual summary $x = x_1, \dots, x_I$ of length I, we model the aligned knowledge base (KB) $\{(k_j, v_j)\}_{j=1}^J$ consisting of keys $k = \{(e_j, t_j)\}_{j=1}^J$, obtained by combining the entity e_j and type t_j in a fact, and values $v = \{v_j\}_{j=1}^J, v_j \in \mathcal{V}$. The KB (k, v) contains J facts, or (k_j, v_j) pairs. Our goal is to locate and extract facts from x by modeling the values v given the text x and the keys k.

Modeling only the KB (k, v) given the text x is not sufficient, as our goal is to locate fact mentions. 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 key pairs in our KB, 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 into a sequence-level decision to resolve conflicts.

Consider the following example: Let our KB consist of the data table and values,

$$D = \left\{ \begin{array}{c} (e_1 = \text{John Doe}, t_1 = \text{POINTS}), \\ (e_2 = \text{John Doe}, t_2 = \text{REBOUNDS}), \\ (e_3 = \text{John Doe}, t_3 = \text{FIRST\_NAME}), \\ (e_4 = \text{John Doe}, t_4 = \text{LAST\_NAME}), \\ \vdots \end{array} \right\}, v = \left\{ \begin{array}{c} v_1 = 8, \\ v_2 = 12, \\ v_3 = \texttt{`John'}, \\ v_4 = \texttt{`Doe'}, \\ \vdots \end{array} \right\}$$

aligned to the summary $x = $ `‘John Doe scored eight points’`. (Give example values of RVs?)

1.1 Model

In order to perform extraction with justification we introduce the following latent variables:

- $m = m_1, \dots, m_I$ where each $m_i \in \{0, 1\}$ indicates whether word x_i is part of a value mention. We aim to identify at least one word in a multiword mention.
- $a = a_1, \dots, a_I$ where each $a_i \in \{1, \dots, J\}$ gives the key k_{a_i} whose value v_{a_i} is mentioned at index i.
- $z = z_1, \dots, z_I$ where each $z_i \in \mathcal{V}$ translates a value mention into the canonical representation of the value as determined by the schema of the KB. For example, the KB may only store numbers in numerical form, as opposed to alphabetical.

We define the model by the following joint distribution over values and latent variables:

$$\begin{aligned} p(v, z, a, m \mid x, k) &= p(v \mid z, a, m, x, k) p(z \mid x) p(a \mid x, k) p(m \mid x) \\ &= \left(\prod_{j=1}^{J} p(v_j \mid z, a, m, x, k) \right) \left(\prod_{i=1}^{I} p(z_i \mid x) p(a_i \mid x, k) p(m_i \mid x) \right) \end{aligned} \tag{1}$$

We proceed to detail the model in two stages. We first give the word-level process, which makes a series of choices for each word x_i in the text $x = x_1, \dots, x_I$, then the sequence-level process which aggregates all of the word-level decisions.

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

1. Value mention identification: Given a sequence of words x, we independently choose whether each word is a value mention with

$$p(m \mid x) = \prod_i p(m_i \mid x) \tag{2}$$

2. Alignment: Each word x_i is independently aligned to a record in the knowledge base with

$$p(a \mid x, D) = \prod_i p(a_i \mid x, D) \tag{3}$$

We align the word x_i by choosing the cell whose value generates the possible value mention at index i. In particular, $a_i = j$ denotes the alignment to the cell d_j. We assume that each value mention aligns to a single record. Although we align every word to a cell, words that are not chosen to be mentions, i.e. x_i such that $m_i = 0$, will be ignored.

3. Translation: All words are translated into a value from the KB schema with

$$p(z \mid x) = \prod_i p(z_i \mid x) \tag{4}$$

with $z_i \in \mathcal{V}$. We would like this to capture synonyms of values not captured by the limited schema of the KB. Although every word is translated to a value, we ignore those that are not chosen as value mentions.

Finally, we aggregate the word-level choices at the sequence level in order to make a single choices for the values v given the summary x.

4. Aggregation: Given the word-level values z, alignments a, mentions m, and data table D we choose the sequence-level values v_j independently from

$$p(v \mid z, a, m, d) = \prod_j p(v_j \mid z, a, m, k_j) \tag{5}$$

If any of the word-level choices disagree, for example if z_1, a_1, m_1 indicates that John scored 19 points, but z_2, a_2, m_2 indicates John scored 18, we must aggregate these choices into a single distribution over values.

We assume each of the word-level choices in each of z, a, m are made independently given the text x and keys k. This gives us the following factorization of the relation extraction system:

1.2 Parameterization

We parameterize the conditional distributions for mention identification, alignment, translation, and aggregation 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 keys k to vectors in $\mathbb{R}^d$.

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

$$p(m_i \mid x) \propto \exp(W_m \mathbf{h}_i)_{m_i}, W_m \in \mathbb{R}^{2 \times d} \tag{6}$$

2. Alignment: We use a bilinear function of the cell embeddings and contextual embeddings to parameterize the alignment distribution.

$$p(a_i \mid x, d) \propto \exp(E(k_{a_i})^T W_d \mathbf{h}_i) \tag{7}$$

with $W_d \in \mathbb{R}^{d \times d}$.

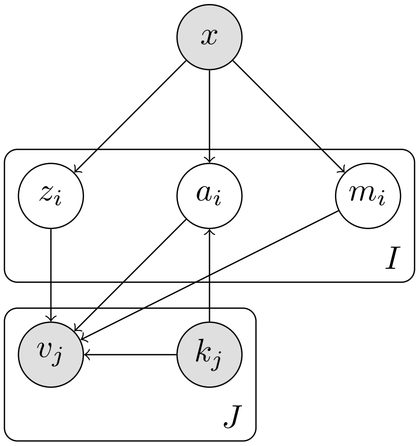

Figure 1: Our model predicts word-level values and alignments then aggregates those choices over all indices i to make a single decision for each value. Each word has the following latent variables: the mention $m_i \in \{0, 1\}$ indicates whether word x_i is a value mention, the alignment a_i gives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

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

$$p(z_i \mid x) \propto \exp(W_z \mathbf{h}_i)_{z_i}, W_z \in \mathbb{R}^{|\mathcal{V}| \times d} \tag{8}$$

4. Aggregation: We parameterize a distribution over the value v_j of cell k_j by counting the 'votes' of relevant word-level values. In order for a word x_i to vote on a value v_j the word must be a mention such that $m_i = 1$, it must be aligned to the correct cell $a_i = j$, and its translation must match $z_i = v_j$. Once the votes are collected, they are aggregated then normalized into a distribution.

   If no word votes for any values for key k_j, i.e. there does not exist a word x_i such that $a_i = j$ and $m_i = 1$, we choose the value given only the cell $p(v_j \mid k_j) \propto \exp(E(v_j)^T W_v [E(k_j)])$.

   Formally, the conditional distribution is given by

$$p(v_j \mid z, a, m, k) \propto \begin{cases} \prod \exp(\psi(v_j, z_i, a_i, m_i, k)), & \exists i, m_i = 1 \land a_i = j \\ \exp(E(v_j)^T W_v [E(k_j)]), & \text{otherwise} \end{cases} \tag{9}$$

$$\psi(v_j, z_i, a_i, m_i, k) = \mathbf{1}(v_j = z_i, a_i = j, m_i = 1) \tag{10}$$

2 Training and Inference

To train a latent variable model, we must marginalize over the unobserved RVs and maximize the likelihood of the observed. Although the latent variables z, a, m are important, as they indicate where and what facts are contained in the text, they are not observed and must be marginalized over. Therefore we would ideally optimize the following objective

$$\log p(v \mid x, D) = \log \sum_{z,a,m} p(v, z, a, m \mid x, D) \tag{11}$$

However, maximizing $\log p(v \mid x, D)$ directly is very expensive for this model as the summation over variables z, a, m is intractable; in particular, the summation has computational complexity $O((|\mathcal{V}| \cdot J \cdot 2)^I)$.

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

2.1 Inference Network

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

$$\log p(v \mid x) \geq \mathbb{E}_{q(z,a,m|v,x,k)} \left[\log \frac{p(v, z, a, m \mid x, k)}{q(z, a, m \mid v, x, k)}\right] \tag{12}$$

Recall that $\mathbf{h}_i \in \mathbb{R}^d$ is a contextual embedding of the word x_i. We introduce the following attention weights over records in order to get a weighted representation of the KB for each index i:

$$\mathbf{g}_j = [E(k_j); E(v_j)] \tag{13}$$

$$\alpha_j \propto \exp(\mathbf{g}_{r_j}^T W_\alpha \mathbf{h}_i) \tag{14}$$

We propose to parameterize the inference network $q(z, a, m \mid v, x, k)$ as follows:

1. Identification: To predict whether x_i is a mention, the inference network uses a weighted representation of the KB along with the contextual representation $\mathbf{h}_i$

   $$p(m_i \mid v, x) \propto \exp\left(V_m \text{MLP}\left(\left[\sum_j \alpha_j \cdot \mathbf{g}_j; \mathbf{h}_i\right]\right)\right)_{m_i}, V_m \in \mathbb{R}^{2\times d} \tag{15}$$

   where MLP is a neural network.

2. Alignment: The alignment model $q(a_i \mid v, x, k)$ uses the attention weights to parameterize the alignment $p(a_i \mid x) = \alpha_{a_i}$.

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

The joint distribution of z, a, m in the inference network is given by

$$\begin{aligned} p(z, a, m \mid v, x, k) &= p(z \mid a, v, x) p(a \mid v, x, k) p(m \mid v, x) \\ &= \prod_i p(z_i \mid a, v, x) p(a_i \mid v, x, k) p(m_i \mid v, x) \end{aligned} \tag{16}$$

We use the score function gradient estimator to perform gradient ascent on the objective in Equation 16 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 pretraining $p(z_i \mid x)$ to ensure that for words x_i where there exists a $z_i \in \mathcal{V}$ such that x_i and z_i are a lexical match we assign high probability to the transliteration $p(z_i = x_i \mid x)$.

3 Evaluation

We are interested in evaluating whether the model can discover and locate the subset of facts in a KB that are expressed in the text without observing the values of the KB. We compare the extractions of our model with a ground truth set of facts extracted by a human annotator, reporting the micro-averaged precision, recall, and F1 score. The grouth truth facts consist of entities, types, values, as well as the location of their value mention in text.

We perform extraction by finding

$$\arg\max_{z_i^*, a_i^*, m_i^*} p(z_i \mid x), p(a_i \mid x, D), p(m_i \mid x)$$

for each word x_i. We obtain the fact (e_j, t_j, v_j) at position i if $m_i^* = 1$, $a_i^* = j$ and $z_i^* = v_j$. The extraction is considered correct if the ground truth contains a fact with the same entity, type, value, and location.