Relation Extraction

Justin T. Chiu

July 31, 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. In order to justify the extraction, we locate the fact's *trigger*, which clearly expresses its occurrence. We make the assumption that a fact's trigger is its value mention.

The problem description is as follows: Given a textual summary $x = x_1, \ldots, x_I$ 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,

$$D = \left\{ \begin{array}{c} k_1 = (e_1 = \text{John Doe}, t_1 = \text{POINTS}), \\ k_2 = (e_2 = \text{John Doe}, t_2 = \text{REBOUNDS}), \\ k_3 = (e_3 = \text{John Doe}, t_3 = \text{FIRST\_NAME}), \\ k_4 = (e_4 = \text{John Doe}, t_4 = \text{LAST\_NAME}), \\ \vdots \end{array} \right\}, v = \left\{ \begin{array}{c} v_1 = 8, \\ v_2 = 12, \\ v_3 = \texttt{`John'}, \\ v_4 = \texttt{`Doe'}, \\ \vdots \end{array} \right\}$$

aligned to the summary $x = \texttt{`John Doe scored eight points'}$. As an example extraction, we identify the value mention `'eight'` as the trigger for the fact mention (k_1, v_1). The fact (k_2, v_2) is not expressed in the text.

1.1 Model

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

- $m = m_1, \ldots, m_I$ 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.
- $a = a_1, \ldots, a_I$ where each $a_i \in \{1, \ldots, J\}$ indicates that word x_i mentions the fact with key k_{a_i}.
- $z = z_1, \ldots, z_I$ where each $z_i \in \mathcal{V}$ translates the value mention containing word x_i 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.

Our model specifies the joint distribution over the values v and latent variables z, a, m given the text x and keys k:

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

The extraction model has three components. The model must identify whether word x_i is a mention using the *identification* model $p(m_i \mid x)$, align word x_i to a key k_{a_i} via the *alignment* model $p(a_i \mid x, k)$, and translate the word x_i into a value z_i with the *translation* model $p(z_i \mid x)$.

This model assumes that each word x_i can explain at most one fact in the KB. Additionally, we make the simplifying assumption that a value mention's translation is unambiguous and independent from the alignment to a key. This limits the expressivity of the model, but allows us to make all choices independently for each word given the text x and keys k.

It is possible for there to be conflicts in the word-level extraction model if two words are mentions with the same alignment but different values. The aggregation model $p(v_j \mid z, a, m, x, k)$ is necessary to resolve these conflicts, and uses the per-word latent variables z, a, m to predict the values of the KB.

We give the parameterizations of all conditional distributions in the following section.

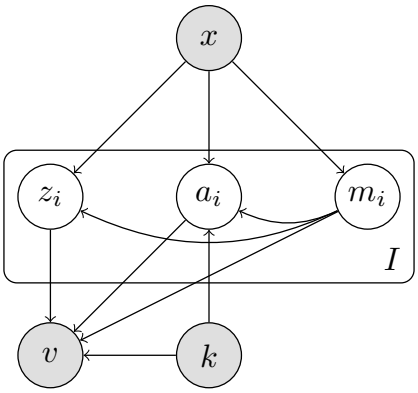

Figure 1: Our model predicts word-level values and alignments then aggregates those choices over all indices i to make a single decision for each value. Each word has the following latent variables: the mention $m_i \in \{0, 1\}$ indicates whether word x_i is a value mention, the alignment a_i gives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

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 predict whether a word x_i is part of a value mention.

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

2. Alignment: We use a bilinear function of the cell embeddings and contextual embeddings to parameterize the alignment distribution. If a word is chosen not to be a mention such that $m_i = 0$, we align $a_i = \texttt{None}$ to a none alignment.

$$\begin{aligned} p(a_i \mid m_i = 1, x, k) &\propto \exp(E(k_{a_i})^T W_a \mathbf{h}_i) \\ p(a_i = \texttt{None} \mid m_i = 0, x, k) &= 1 \end{aligned} \tag{3}$$

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

3. Translation: We use the contextual embedding to translate a word into a value. If a word is chosen not to be a mention such that $m_i = 0$, we set z_i to a none token.

$$\begin{aligned} p(z_i \mid m_i = 1, x) &\propto \exp(W_z \mathbf{h}_i)_{z_i}, W_z \in \mathbb{R}^{|\mathcal{V}| \times d} \\ p(z_i = \texttt{None} \mid m_i = 0, x) &= 1 \end{aligned} \tag{4}$$

4. Aggregation: We choose the value associated with a particular key in the KB by first using the translated values of any mentions aligned to the corresponding key. If no

mention is aligned to the key, we instead ignore the text and choose the value given only the key.

Formally, the conditional distribution is given by

$$p(v_j \mid z, a, m, k) \propto \begin{cases} \prod_i \exp(\psi(v_j, z_i, a_i, m_i)), & \exists i, m_i = 1 \wedge a_i = j \\ \exp(E(v_j)^T W_v[E(k_j)]), & \text{otherwise} \end{cases} \tag{5}$$

$$\psi(v_j, z_i, a_i, m_i) = \mathbf{1}(v_j = z_i, a_i = j, m_i = 1) \tag{6}$$

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

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

$$\log p(v \mid x, k) = \log \sum_{z,a,m} p(v, z, a, m \mid x, k) \tag{7}$$

However, maximizing $\log p(v \mid x, k)$ directly is very expensive 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:

$$\log p(v \mid x) \geq \mathbb{E}_{q(z,a,m|v,x,k)} \left[\log \frac{p(v, z, a, m \mid x, k)}{q(z, a, m \mid v, x, k)} \right] \tag{8}$$

We use the same contextual embedding $\mathbf{h}_i \in \mathbb{R}^d$ of the word x_i from the extraction system, and we extend the embedding function E to embed not only keys but also values. We introduce the following attention weights over records in order to get a weighted representation of the KB for each index i:

$$\begin{aligned} \mathbf{g}_j &= [E(k_j); E(v_j)] \\ \alpha_j &\propto \exp(\mathbf{g}_{r_j}^T W_\alpha \mathbf{h}_i) \end{aligned} \tag{9}$$

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 $\mathbf{h}_i$

$$p(m_i \mid v, x) \propto \exp\left(V_m \text{MLP}\left(\left[\sum_j \alpha_j \cdot \mathbf{g}_j; \mathbf{h}_i\right]\right)\right)_{m_i}, V_m \in \mathbb{R}^{2\times d} \tag{10}$$

   where MLP is a neural network.

2. Alignment: The alignment model $q(a_i \mid v, x, k)$ uses the attention weights in Eqn. 9 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

$$\begin{aligned} p(z, a, m \mid v, x, k) &= p(z \mid a, v, x)p(a \mid v, x, k)p(m \mid v, x) \\ &= \prod_i p(z_i \mid a, v, x)p(a_i \mid v, x, k)p(m_i \mid v, x) \end{aligned} \tag{11}$$

We use the score function gradient estimator to perform gradient ascent on the objective in Equation 11 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 $m_i = 0$ for every word x_i. 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

$$\arg\max_{z_i, a_i, m_i} p(z_i \mid x), p(a_i \mid x, k), p(m_i \mid x)$$

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.