A Variational Approach to Knowledge Graph Node Embeddings

July 26, 2017

Abstract

We present a Variational approach to to the problem of embedding knowledge graph nodes.

1 Introduction

Knowledge graph node embeddings are vector representations of the nodes in a knowledge graph. For a knowledge graph with nodes N, an embedding is a function $v : N \to \mathbb{R}^k$ such that v organizes the nodes meaningfully in the spacer $\mathbb{R}^k$ according to some criterion.

Effective embeddings are an important first step in many pipelines for knowledge graph visualisation, completion, classification and other tasks. Many embedding algorithms model the relations in the knowledge graph as maps in the embedding space: either translations, or more complex transformations. If for instance, we have the triple $(\texttt{Mary}, \texttt{childof}, \texttt{John})$, then we learn embeddings $v_{\texttt{Mary}}, v_{\texttt{John}} \in \mathbb{R}^k$ (for some chosen embedding dimension k) and a map $f_{\texttt{childof}} : \mathbb{R}^k \to \mathbb{R}^k$ for the relation `childof` such that $f_{\texttt{childof}}(v_{\texttt{John}})$ should be as close to $v_{\texttt{Mary}}$ as possible (for some chosen distance metric).

This scheme works well, but has trouble representing multiple relations. If John has more than one child, what should $f_{\texttt{childof}}(v_{\texttt{John}})$ be? Many approaches have been introduced, often requiring an advanced mathematical structure for the embedding space, such as complex vectors [**?**], or holographic projections [**?**].

Our approach is based on a simple intuition. For each source node, we place a spherical Gaussian distribution over the embedding. Transforming this Gaussian by the relational map should then result in a new Gaussian

which maximizes the likelihood of the nodes connected to the original by the given relation.[1] In other words, transforming a spherical Gaussian centered on $v_{\texttt{John}}$ by $f_{\texttt{childof}}$ should result in a distribution that maximizes the probability of John's children.

Intuitively, this is a simple latent variable problem, that can be solved by an alternating optimization algorithm in the vein of expectation-maximization: given a choice for the node embeddings, we can find the optimal functions to represent the relations, and given a choice for the relation functions, we compute the optimal embeddings. Alternating these two optimizations is likely to converge to a good solution. To formalize this intuition in an elegant manner, take a Bayesian approach, and find the optimum using variational inference. Thge preicsae model is detailed in Section 2.

Our approach has several advantages over the alternatives:

- The nodes are embedded in a simple Euclidean space, making no special demands on downstream architectures.
- The multiple-relation issue is solved in an intuitive manner: mapping a parent node to its children creates a probability distribution that covers the children.
- The method is agnostic to the family of maps used to represent relations. We choose affine maps as a good tradeoff between ease of optimization, rigidity to protect against overfitting, and expressiveness. The method easily generalizes to other function classes.
- For linear maps, we can decompose the loss function into factors which can be solved analytically. Thus, the problem can be solved using an iterative algorithm which is guaranteed to converge to a local optimum.
- Negative sampling is not required. The basic framework constrains the problem sufficiently to lead to models that generalize well.
- The variational framework protects against convergence to solutions with singularities, and allows us to incorporate hyperparameters like the dimension of the embedding space into the optimization process.

We test our embeddings on the tasks of knowledge base completion and node classification. We outperform the state-of the art in each case. We also compare a general-purpose stochastic gradient descent approach for our

[1] This approach is inspired by the Coherent Point Drift algorithm [], which uses a similar approach for the problem of point set registration.

model, to the variational search, showing that the latter converges faster, in fewer iterations, and to better solutions.

2 Model

Define a knowledge graph G as a triple $G = (V_G, R_G, E_G)$ with V_G representing a set of labeled nodes, R_G representing a set of available relations, and $E_G \subset V_G \times R_G \times V_G$ the set of *triples*—edges labeled with relations. Since we are not interested in modeling the internal structure of labels, we will assume that V is always the set of the first $|V|$ natural numbers, and likewise for R.

To simplify notation, we will assume that the graph being modeled is clear from context, and drop the subscript G.

Let $x_i \in \mathbb{R}^k$ be the embedding chosen for node $i \in V$. Let f_r be a parametrized function from $\mathcal{F}$, representing relation $r \in R$. For our method, we will let $\mathcal{F}$ be the set of affine functions, each parametrized by a transformation matrix $A_r \in \mathbb{R}^{k \times k}$, and a translation vector $t_r \in \mathbb{R}^k$. Thus our model contains $|N|$ vectors in $\mathbb{R}^k$, and $|R|$ functions each defined by a matrix A_r and vector t_r.

To formalize the intuitions expressed in the introduction, we first express the constraints in a graphical model. Let $E^r_{\text{subject}} = \{s \mid \exists o, (s, r, o) \in E\}$ and $E^{r,o}_{\text{object}} = \{s \mid (s, r, o) \in E\}$. Figure **??** shows the relations between the different random variables. Interestingly, this model contains *no observed variables*. The data is used purely to inform the topology of the graphical model.[2]The model also gives us the priors: the prior probability for the embeddings, for which we will use the standard normal distribution: $\mu = 0$ and $= \mathbf{1}$ and the prior W_t on the

[3]

$$\underset{\{x_i\},\{f_r\},\sigma}{\arg\min} \prod_{(s,r,o)\in E_G} f_r(N_{x_s,\sigma})(x_o)$$

3 Experiments

4 Conclusion

[3]Modeling the data as observations would lead to tensor-factorization approaches like ReSCAL [].