One Semantic parser base on Seq2Seq in MSParS

Shengqiang Zhang   Huiqiang Jiang   Yixu Wang   Jiangyong Wu
Peking University
{1801210730, 1801210840, 1801210668, 1801210679}@pku.edu.cn

Abstract

Semantic parsing is a process to obtain structure representations. In this work, we propose a deep-learning model that competed at DatsSet MSParS "a Multi-perspective Semantic ParSing Dataset for Knowledge-based Question Answering". We used Sequence to Sequence(Seq2Seq), on encoder end using a pre-trained word embeddings on a big collection of Wikipedia 2014 + Gigaword 5 of glove.6B.100d. We achieve a 65.63% BLEU score on dev, accuracy is 84.7%.

1 Introduction

Semantic parsing is a process to map natural language utterances onto machine interpretable meaning representations. There has recently been a surge of interest in developing machine learning methods for semantic parsing, due in part to the existence of corpora containing utterances annotated with formal meaning representations. In order to predict the correct logical form for a given utterance, most previous systems rely on predefined templates and manually designed features, which often render the parsing model domain or representation-specific. In this work, we aim to use a simple yet effective method to bridge the gap between natural language and logical form with minimal domain knowledge. And we experiment in an open-domain Dataset, MSParS.(**?**)

Encoder-decoder architectures based on attention networks have been successfully applied to a variety of NLP tasks ranging from summarization (**??**) , to machine translation (**?**). In this work, we adopt the general encoder-decoder paradigm to the semantic parsing task. Our model is base on Sequence to Sequence model (**?**) and it learns from natural language descriptions paired with meaning representations; it encodes sentences and decodes logical forms using recurrent neural networks with attention units.

2 Our work

Our goal is to learn semantic parsers from instances of natural language expressions paired with their structured meaning representations.

Let $x = x_1 \cdots x_{|x|}$ denote a natural language expression, and $y = y_1 \cdots y_{|y|}$ its meaning representation. We wish to estimate $p\left(y|x\right)$, the conditional probability of meaning representation y given input x. We decompose $p\left(y|x\right)$ into a two-stage generation process:

$$p\left(y|x\right) = p\left(y|x,a\right) p\left(a|x\right) \tag{1}$$

2.1 Encoder-Decoder

An *encoder* is used to encode the natural language input x into vector representations. Then, a *decoder* learns to compute $p\left(a|x\right)$ and generate the sketch a conditioned on the encoding vectors.

Input Encoder Every input word is mapped to a vector via $\mathbf{x}_t = \mathbf{W}_x \mathbf{o}\left(x_t\right)$, where $\mathbf{W}_x \in \mathbb{R}^{n \times |\mathcal{V}_x|}$ is an embedding matrix, $|\mathcal{V}_x|$ is the vocabulary size, and $\mathbf{o}\left(x_t\right)$ a word embedding vector. We use a bi-directional recurrent neural network with gated recurrent unit as the input encoder.

2.2 Attention Structure

Our encoder and decoder are tied to each other through a multi-hop attention mechanism(**??**). For each decoder layer ℓ, we compute the attention a_{ij}^{ℓ} of state i and source element j as:

$$a_{ij}^{\ell} = \frac{\exp(d_i^{\ell} \cdot z_j^u)}{\sum_{t=1}^{m} \exp(d_i^{\ell} \cdot z_t^u)}, \tag{2}$$

where $d_i^{\ell} = W_d^{\ell} h_i^{\ell} + b_i^{\ell} + g_i$ is the decoder state summary combining the current decoder state h_i^{ℓ} and the previous output element embedding g_i. The vector $\mathbf{z^u}$ is the output from the last encoder layer u. The conditional input c_i^{ℓ} to the current

Corups	BLEU	Accuracy	loss
TrainSet	**66.08**	**93.0**	**0.383**
DevSet	65.63	84.7	1.39

Table 1: Comparison between different Corous using Seq2Seq on MSParS (in %)

decoder layer is a weighted sum of the encoder outputs as well as the input element embeddings e_j:

$$c_i^\ell = \sum_{j=1}^{m} a_{ij}^\ell (z_j^u + e_j). \tag{3}$$

The attention mechanism described here performs multiple attention "hops" per time step and considers which words have been previously attended to. It is therefore different from single-step attention in recurrent neural networks (**?**), where the attention and weighted sum are computed over $\mathbf{z^u}$ only.

3 Experiments

In this section, we evaluate in train, dev set. We use glove.6B.100d as our word embedding. And train our model for 140 epochs. We test our model in MSParS. (**?**)

The MSParS dataSet is a QA set. From our understanding, in this dataSet, you should extract Named entities. And then it also need to find relation of this entities.

After experiments, we get 66.08% BLEU, 93.0% accuracy in train set, and 65.63% BLEU, 84.7% accuracy in dev set.

4 Conclusion

In this paper, we introduce a deep-learning sematic parser model. Glove is the best way in our jobs on the embedding layer. And we test our model in MSParS DataSet. We achieve a 66.08% BLEU, 93.0% accuracy in train set, and 65.63% BLEU, 84.7% accuracy in dev set. After our work, we only test our model in MSParS DataSet. In the future, We can do more work in word representations, beam search, parser analysis.

Algorithm 1 match SHIPINFO and HEALTHINFO V1

Input: e: ele shipInfo (name, address)
h: healthInfo (name, address)
$e.name$: name $\Rightarrow$ (tradeMark,subbranch)
Output: r: Match result
function MATCHELEV1(e, h)
  if $e.name$ match $h.name$ **then** ▷ *match* $\neq$ *equal*
    return $\arg\max(\text{LCS}(e.address, h.address))$
  if $e.address$ match $h.address$ **then**
    return $\arg\max(\text{LCS}(e.name, h.name))$
  // if not match, **do word segment,**
  $e.nameSeg = seg(e.name)$,
  $h.nameSeg = seg(h.name)$,
  $e.addrSeg = seg(e.address)$,
  $h.addrSeg = seg(h.address)$
  if match($e.nameSeg, h.nameSeg$) **and match(**$e.addrSeg$**,** $h.addrSeg$**) then**
    return $\arg\max$**(LCS(**$e.address$**,** $h.address$**))**
  return r