Semantic Parsing with Encoder-Decoder Models

Anonymous ACL submission

Abstract

In this project, I have implemented Encoder-Decoder Models to perform semantic parsing. In this task, a given natural language sentence is converted to the formal representation such as as Lambda calculus. Since the sentence would be fully disambiguated, they could be treated as a source code and processed with a knowledge base. For this task, I used the given GeoQuery dataset(Zelle and Mooney, 1996). First, I created an Encoder-Decoder model with GRU layers. For the second part, I used attention mechanism in the Encoder-Decoder model with GRU layers. Finally, for the extension part I implemented a training improvement technique, scheduled sampling. Based on the (Bengio et al., 2015), I used scheduled sampling with linear, exponential and inverse sigmoid decay and compared the training improvement using these decay schemes. I collaborated with Neeharika Immaneni and Sundar for this project.

1 Introduction

In this project, I used an encoder-decoder model for a semantic parsing task. Given a natural language sentence, this system would convert the input to lambda calculus representation such that, this could be treated as a source code. I used the Geoquery dataset[ref] that consist of sentences such as What is the population of Atlanta ga?, what state border Texas?. For each of these cases, answer would be computed by executing the expression against the knowledge base. The data consist of 480 training examples and 120 dev examples. Using the train data, an input and output indexer is created of size 238 and 153 respectively. The maximum length of the sentence in the input data is 19 and the maximum length of output sentence is 65. The evaluation of these models are done either using the token level accuracy or the logical form comparison or the denotation based accuracy where, the answer which the logical form would produce when when executed against the knowledge base. This project is divided into three parts. In the first part, a simple encoder-decoder based model using GRU layers is used for the semantic parsing task. In the second part, the attention mechanism is incorporated into the encoder-decoder model which leads to improved performance. Finally, for the extension part I used scheduled sampling with linear, exponential and sigmoid decay. Scheduled sampling helps in training the decoder with the predictions rather than feeding the gold label for all tokens. This would help in bridging the gap between training and inference where, the gold label is never available during inference for the network.

2 Part 1: Encoder-Decoder Model

The encoder part of the network is made of an embedding layer and various sized of hidden GRU layers. The embedding layer has a dimension of 238. The output of the embedding layer is fed to the GRU layer and then finally to its output is passed to the decoder. The model is trained for using a single sample, i.e., batch size 1 using the Negative Log Likelihood criterion. The encoder is fed with one token at a time and the output is stored in an encoder_outputs tensor. The decoder is also fed with one token at a time for a maximum token length of 65. For the first token, the decoder is fed with the SOS token as the decoder input and for the following tokens, it is fed with the true label as the decoder input. The reason being, I used the teacher forcing algorithm where, during training, the decoder will be fed with the actual label for improved learning. Finally, if the decoder produces an EOS_token or POS token, the decoder will stop taking inputs for that particular sentence. This way, the moment EOS or POS is reached for a padded sentence, the decoder is not trained further. Dur-

ing evaluation, the encoder_net, decoder_net along with the hidden units as parameters is passed for the evaluation module. The encoder takes the indexed sentence as one token at a time as input and the final encoder hidden state is fed as the input to the decoder module. The decoder is fed with individual tokens and the hidden state. The first decoder input is the SOS token and for the rest of the tokens in the sentence, the decoder is fed with the previous tokens topi prediction. As soon as an EOS or POS token is predicted, the decoding stops and the next input sentence is passed to the encoder. The encoder was designed with an embedding layer, a gru with 512 hidden size. The decoder layer has a embedding layer, followed by ReLU activation, gru with 512 hidden units and finally a log softmax that predicts between 153 classes. I set the learning rate to 0.001 and used Adam optimizer for a negative log-likelihood loss criterion. I used batch size of 1 for training the model with xavier initialization of gru parameters. Using this encoder-decoder model with a teacher forcing ratio of 0.5, i.e., for every example in the train data, a random probability is generated and if the random probability is greater than 0.5, the decoder is fed with the gold label. Otherwise, the decoder is fed with the predicted label. To speed-up the training process, denotation accuracy was calculated only on the dev dataset. This resulted in a 86.8% token-level accuracy and 42.7% exact logical form matches on the train. On the dev dataset, after 20 epochs, 62.2% token-level accuracy, 16.7% exact logical form match and 25.0% denotation matches.

3 Part 2: Encoder-Decoder with Attention

The encoder and decoder setup is created similar to part 1. In this part, the attention is included in the decoder model. In part 1, when the context vector is passed between the encoder and the decoder, it has to carry all the information about the entire sentence, which may not be effective. Therefore, using the attention mechanism, it will help the decoder to focus on different parts of the encoders output for every step of the decoders output. The attention weights are calculated using a feed-forward layer. The attention applied input is then passed to the GRU layer, followed by a log_softmax to predict the output token class. Similar to the implementation given, I used the Bahdanau attention mechanism. The encoder was designed with a embedding layer, followed by a gru layer of 512 hidden units. The decoder was designed with linear layer that takes the attention weights for computation followed by gru with 512 hidden units and a log softmax activation. I used the Adam optimizer with 0.001 learning rate, batch size 1 for training with negative log-likelihood loss criterion. Similar to the encoder-decoder model, I used teacher forcing ratio of 0.5 while training the decoder.

4 Extension

In order to bridge the gap between training a testing, I implemented scheduled sampling algorithm. In this algorithm, instead of always training the decoder with the gold label and using its predictions while inference, which could accumulate the error generated by the network, an alternate between training on gold label vs. decoder prediction is utilized. Using the exponential, linear and inverse sigmoid decay functions, the decoder will be trained with gold label with a higher probability in the start and then gradually, this probability will be reduced as the model learns better. The hypothesis is that, initially the model would require larger teacher forcing to make appropriate predictions and as the training progresses, the decoder's output would be more reliable and can be used as input for the next token while training. I used this algorithm for both Encoder-Decoder and attention based Encoder-Decoder models with three different decay schemes. In the encoder decoder model, exponential decay have the best results when the models were trained for 20 epochs. Using the exponential decay scheduled sampling, the denotation match was 35.0. This shows that for the encoder-decoder model, scheduled sampling with exponential decay performed better than other decay or no decay schemes. With the exponential decay, teacher forcing is withdrawn almost immediately after training starts and I believe this helps the model in learning better and the gap between the training and inference is better filled since teacher forcing is not ultimately scraped out but it is used for a fewer iterations of training compared to other decay methods. In the attention model, I ran the training for 50 epochs and noted epoch at with the best dev accuracy. The teacher forcing ratio set at 0.5 gave the best dev accuracy after training for 44 epochs. Among the three decay schemes, inverse sigmoid performed better. This could be because, the attention already takes care of retaining the in-

formation from the encoder and teacher forcing at the decoder is helpful only when it is higher at the beginning and linearly decreased, with a lower decay rate towards the end.

5 Conclusion

In this project, I used encoder-decoder and attention based encoder-decoder model for semantic parsing. The attention based encoder-decoder performed better than the encoder-decoder model. For the extension part, I used scheduled sampling with three types and decay and found that exponential decay performed better for encoder-decoder model compared to teacher forcing with 0.5 and other decay schemes. In attention based encoder-decoder model, inverse sigmoid decay performed better than the other decay schemes.

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, pages 1050–1055. AAAI Press.

Model	**Scheduled Sampling**	**Result-Train** [token-level, exact match]	**Result-dev** [token-level, exact match, denotation]
Enc-Dec	No Decay	86.8, 42.7	62.7, 16.7, **25.0**
Enc-Dec	Linear	98.8, 67.3	64.3, 16.7, 28.3
Enc-Dec	Exponential	99.1, 75.8	67.2, 28.3, **35.0**
Enc-Dec	Inv sigmoid	79.4, 38.8	57.1, 11.7, 20.8
Atten Enc-Dec	No Decay	97.8, 79.4 (44 epochs)	74.3, 39.2, **51.7**
Atten Enc-Dec	Linear	99.3, 81.5 (32 epochs)	70.3, 26.7, 40.0
Atten Enc-Dec	Exponential	99.5, 84.4 (38 epochs)	73.7, 35.8, 45.8
Atten Enc-Dec	Inv sigmoid	93.9, 73.1 (42 epochs)	74.5, 37.5, 49.2

Table 1: The result on the train and dev dataset for encoder-decoder and attention based encoder-decoder models are given.