# Transferring Knowledge from Structure-aware Self-attention Language Model to Sequence-to-Sequence Semantic Parsing

**Anonymous ACL submission**

## Abstract

Semantic parsing considers the task of mapping a natural language sentence into a target formal representation, where various sophisticated sequence-to-sequence (seq2seq) models have been applied with promising results. Generally, these target representations follow a syntax formalism that limits permitted forms. However, it is neither easy nor flexible to explicitly integrate this syntax formalism into a neural seq2seq model. In this paper, we present a structure-aware self-attention language model to capture structural information of target representations and propose a knowledge distillation based approach to incorporating the target language model into a seq2seq model, where grammar rules, sketches or extra corpus are not required in the training process. An ablation study shows that the proposed language model can notably improve the performance of the baseline model. The experiments show that our method achieves new state-of-the-art performance among neural approaches on four semantic parsing (ATIS, GEO) and Python code generation (Django, CoNaLa) tasks.

## 1 Introduction

Semantic parsing aims to map a natural language sentence into a machine executable formal representation, which has been considered as one of the prime challenges nowadays in natural language processing (NLP). These target formal representations can generally be divided into three categories (Kamath and Das, 2018), i.e., logical forms, like first order sentences or $\lambda$-calculus expressions (Zettlemoyer and Collins, 2005), programming language statements, like Python code or SQL programs, and graph-based forms, like labeled graphs in Abstract Meaning Representation (AMR) (Banarescu et al., 2013). In this paper, we focus on semantic parsing that yields logical forms.

Target logical forms often follow a syntax formalism that limits permitted formulas, which can be used to filter the output and improve the performance of semantic parsing. For example, in the pre-neural era, CCG based approaches (Kwiatkowski et al., 2013) achieved significant performance gains by introducing a linguistically motivated grammar induction scheme. Some neural semantic parsers (Yin and Neubig, 2018; Sun et al., 2020) first transduce the natural language utterance into an Abstract Syntax Tree (AST), then serve it as an intermediate meaning representation to incorporate with grammar rules for the target logical form. Semantic parsing can also be considered as a seq2seq transduction problem, where the decoder can leverage structural features of target representations. In particular, hierarchical tree decoders are applied in (Dong and Lapata, 2016; Alvarez-Melis and Jaakkola, 2017; Sun et al., 2019) to take into account the tree structure of the logical expression. Decoders constrained by a grammar model are applied in (Xiao et al., 2016; Yin and Neubig, 2017; Krishnamurthy et al., 2017; Dong and Lapata, 2018). The uncertainty-driving adaptive decoding is used to guide the decoder in (Zhang et al., 2019). Relatively sizeable monolingual corpus of the target programming language is used in (Norouzi et al., 2021) to improve performance.

Note that, manually specified grammar rules, sketches and extra corpus for target logical forms are required in most of these approaches, which limits their adaptabilities and scalabilities to a new semantic parsing task with updated target logical forms. In this paper, we consider using a structure-aware language model to capture formal patterns for target representations and incorporating the language model into seq2seq models for semantic parsing.

We first train the structure-aware language model on target logical forms to capture structural information. Then, we incorporate the language model to a seq2seq model for semantic parsing.

Integrating a language model into a seq2seq model has been considered in automatic speech recognition (ASR) and neural machine translation (NMT). In particular, shallow fusion and deep fusion (Gulcehre et al., 2015) are two such approaches in NMT. Cold fusion (Sriram et al., 2018) is tested on ASR tasks. (Bai et al., 2019) proposes a knowledge distillation based training approach to transferring knowledge from a language model to a seq2seq model for ASR. Here, we follow the knowledge distillation structure to integrate the language model to the baseline seq2seq model for semantic parsing.

We evaluate our approach on two semantic parsing datasets, ATIS (Dahl et al., 1994) and GEO (Zelle and Mooney, 1996) datasets, where target logical forms are $\lambda$-calculus expressions and two code generation tasks, Django(Oda et al., 2015) and CoNaLa(Yin et al., 2018), where target logical forms are Python code. We train the target language model based on target logical forms appeared in the training sets of each datasets without involving extra corpus. The experimental results show that our approach achieves state-of-the-art performance among neural network based approaches on ATIS, GEO, Django and CoNaLa datasets.

In this paper, we show that the proposed language model can be used to capture structural features of target logical forms and the knowledge distillation structure can be used to transfer knowledge to a seq2seq model for semantic parsing, where manually specified grammar rules or sketches, or extra corpus are no longer required. Notice that, this approach can be applied to various sophisticated seq2seq models, which results a more flexible and scalable method for neural semantic parsers to leverage structural features of target representations. The main contributions of the paper are summarized as follows:

- We propose a structure-aware self-attention language model to capture structural information of target logical forms.
- We propose a knowledge distillation structure to transfer knowledge from target language model to a seq2seq model, which suggests a more flexible and scalable method for neural semantic parsers to leverage structural features of target representations.
- We implement the approach on baseline seq2seq models, which achieves new state-of-the-art performance among neural semantic parsers on ATIS, GEO, Django and CoNaLa datasets.

## 2 Related Work

### 2.1 Neural Semantic Parsing

Neural semantic parsing has achieved promising results in recent years. In particular, AST based parsers (Yin and Neubig, 2018; Sun et al., 2020, 2019) first map a nature language sentence into an abstract syntax tree (AST), then parse the AST to the corresponding target logic form. On the other hand, seq2seq based semantic parsers often leverage structural features of natural language sentences or target representations to improve the performance. Specifically, a sequence-to-tree (seq2tree) model (Dong and Lapata, 2016) updates the decoder into a hierarchical LSTM tree, which helps the model to utilize the hierarchical structure of logical forms. A graph-to-sequence (graph2seq) model (Xu et al., 2018) updates the encoder into a graph encoder. Graph neural networks (GNNs) are also used in semantic parsing (Shaw et al., 2019) to incorporate information about relevant entities and their relations during the parsing. A sequence-to-action (seq2action) model (Chen et al., 2018) considers semantic parsing as an end-to-end semantic graph generation process. A coarse-to-fine (coarse2fine) model (Dong and Lapata, 2018) decomposes the decoding process into two stages. The first stage predicates a rough sketch of the meaning representation and the second stage fills in missing details conditioning on the natural language input and the sketch itself. The AdaNSP model (Zhang et al., 2019) proposes an adaptive decoding method to avoid intermediate representations in the parsing process, where the decoder is guided by the model uncertainty. TAE (Norouzi et al., 2021) exploit a relatively sizeable monolingual corpus of the target programming language to improve performance.

Notice that, manually specified grammar rules or sketches, or extra corpus are required in most of these neural semantic parsing approaches to leverage structural features of natural language sentences or target representations. In this paper, we consider using the proposed target language model to capture these formal patterns and incorporating the language model into seq2seq models for semantic parsing.

### 2.2 Integrating Language Model into Seq2Seq Models

Integrating a language model into a seq2seq model has been considered in multiple NLP tasks, like automatic speech recognition (ASR) and neural machine translation (NMT). In particular, shallow fusion and deep fusion (Gulcehre et al., 2015) are proposed to integrate a language model into a seq2seq model. Both methods first train a language model and a translation model separately, then use the language model in the inference step. Specifically, shallow fusion performs a log-linear interpolation between the decoder and the language model to re-weight the translation model's scores during the beam search. Deep fusion concatenates the language model and decoder's hidden states next to each other, then uses the the hidden states to fine-tune the model. Cold fusion (Sriram et al., 2018) is tested on AST tasks. Cold fusion uses the logic outputs of the trained language model as features to train the translation model. Simple fusion (Stahlberg et al., 2018) uses the output of a trained language model together with the output of a translation model to train the translation model. Component fusion (Shan et al., 2019) first trains a source language model, later freezes the source language model and trains the translation model, then replaces the source language model with a target language model in the inference process.

The LST (Learning Spelling from Teachers) approach (Bai et al., 2019) proposes a knowledge distillation based training approach to transferring knowledge from a language model to a seq2seq model for ASR. It first trains a recurrent neural network based language model (RNNLM) on large scale external text, then considers the RNNLM as the teacher to generate soft labels of speech transcriptions to train the decoder in the seq2seq model.

In this paper, we follow the knowledge distillation structure to transfer knowledge from target language model to the decoder of a baseline seq2seq model for semantic parsing. Different from LST, a new Transformer-based structure-aware language model is considered here, which can capture structural information of formal patterns for target representations, and the language model is trained only on target logical forms in the training set of the datasets, where extra corpus is not required. We show that the approach achieves new state-of-the-art performance on ATIS, GEO, Django and CoNaLa datasets.

## 3 Preliminaries

### 3.1 A Seq2Seq Model for Semantic Parsing

The training procedure of a baseline seq2seq model for semantic parsing is illustrated in Figure 1. The parsing model maps natural language sentences into target expression.

First, a natural language sentence is preprocessed into a sequence of word indexes $\mathbf{x} = \{x_1, \ldots, x_m\}$ and the labeled logical form is preprocessed into a sequence of word indexes $\mathbf{y}^* = \{y_1^*, \ldots, y_n^*\}$. Then, the encoder network produces the sequence $\mathbf{x} = \{x_1, \ldots, x_m\}$ into a high level contextual representation $\mathbf{h} = \{h_1, \ldots, h_m\}$. Later, the decoder network generates the target output $\mathbf{y} = \{y_1, \ldots, y_n\}$ from $\mathbf{h}$.

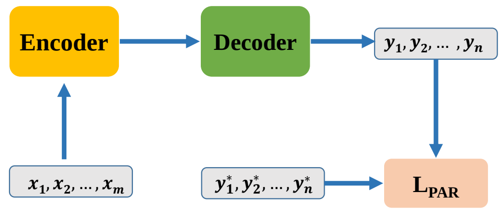



Figure 1: A basic seq2seq model for semantic parsing.

At time step $t$, current token $y_t$ is generated by the following equation:

$$P_{PAR}(y_t) = p(y_t \mid y_{<t}, \mathbf{x}), \tag{1}$$

where $y_{<t} = y_1 \ldots y_{t-1}$, $\mathbf{x}$ represents the input word indexes.

The training criterion is cross entropy:

$$L_{PAR} = -\sum_{i=t}^{T} \sum_{k=1}^{|V|} \mathbf{1}\{y_t = k\} \log P_{PAR}(y_t = k) \tag{2}$$

where $P_{PAR}$ is computed from Equation (1), $T$ is the length of the target sequence, $|V|$ is the size of the vocabulary, $\mathbf{1}$ is the indicator function.

### 3.2 Self-Attention

The multi-head self-attention module is a key component in Transformer (Vaswani et al., 2017). In particular, transformer's sub-layers employ $h$ attention heads to perform self-attention. The results from each attention heads are concatenated and transformed to form the output of the sub-layer.

Given a sequence $x = (x_1, \ldots, x_n)$ as input, each attention head uses scaled dot-product attention to compute a new sequence $z = (z_1, \ldots, z_n)$ of the same length, i.e.,

$$z_i = \sum_{j=1}^{n} \alpha_{ij} \left(x_j W^V\right), \tag{3}$$

where $W^V$ is a matrix of parameters and $\alpha_{ij}$ are normalized by a softmax function, i.e.,

$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{n} \exp\left(e_{ik}\right)}, \tag{4}$$

where $e_{ij}$ is computed using a compatibility function that compares two input elements, i.e.,

$$e_{ij} = \frac{\left(x_i W^Q\right)\left(x_j W^K\right)^\top}{\sqrt{d_z}}, \tag{5}$$

where $W^Q, W^K$ are parameters to be learned.

# 4 Method

In this section, we specify details of our method, i.e., using a knowledge distillation based structure to transfer knowledge from a structure-aware target language model to a seq2seq model. We first introduce the architecture of the new model. Then, we describe the proposed target language model. At last, we provide details of the method in the training process.

## 4.1 Model Overview

An overview of the new model's architecture is shown in Figure 2. Note that, the new model is generated from the basic seq2seq model in Figure 1 by introducing a knowledge distillation structure where the pretrained structure-aware language model serves as the teacher to guide the parsing model.

In specific, the structure-aware language model is pre-trained on target logical forms. The language model contains a structure-aware self-attention transformer encoder to explicitly capture the structural information. It is used to provide soft labels as prior knowledge to "teach" the parsing model in the training process, where the Kullback-Leibler divergence between estimated probabilities is intended to be minimized.

Notice that, there is no specific requirement for the seq2seq model in the architecture. Then, besides the basic seq2seq model, this knowledge distillation structure can be applied to other sophisticated seq2seq models to leverage structural features of target representations.

## 4.2 Target Language Model

Here we specify details of the proposed target language model, i.e., structure-aware self-attention language model. Architecture of the language model is shown in Figure 3.

Since the target logical forms can all be seen as bracket trees, they're tree-structured. Self attention in Transformer learns how much attention to put on words in a sequence, but it ignores the syntactic information of trees. The siblings of tree nodes may have long distance in a sequence position, but they're related closely. Therefore, we propose structure-aware self-attention encode the depth information of sibling nodes into self-attention to capture this information.

Motivated by (Shaw et al., 2018), we extend the self-attention architecture to explicitly encode the relation between an element pair $(x_i, x_j)$ by modifying Equation (5) to

$$e_{ij} = \frac{x_i W^Q \left(x_j W^K + a_{ij}^K\right)^\top}{\sqrt{d_z}}. \tag{6}$$

Different from (Shaw et al., 2018), we redefine the relation representations $a_{ij}$.

We assume that the depth information is less useful when it is too deep. We define the maximum $s$ as a constant $k$:

$$\begin{aligned} a_{ij}^K &= w_{clip(s(i,j),k)}^K, \\ clip(x, m) &= \min(m, x), \end{aligned} \tag{7}$$

where $s(i, j)$ is defined as follows:

$$s(i,j) = \begin{cases} dep(i), & father(i) = father(j), \\ 0, & otherwise, \end{cases} \tag{8}$$

where $dep(i)$ is the depth of node $i$ in a tree, $father(i)$ means the father of node $i$.

Figure 4 shows an example we chose in GEO dataset for demonstration.

We replace the original self-attention architecture of transformer encoder with our structure-aware self-attention. The encoder is bidirectional, so we add the subsequent mask (originally applied in the transformer decoder) to it to specify it as a language model. The subsequent mask creates a diagonal matrix where the elements above the diagonal will be modified to zero and the elements below the diagonal will be set to whatever the input tensor is. Therefore, the prediction for position $i$ will depend only on the known outputs at positions less than $i$.

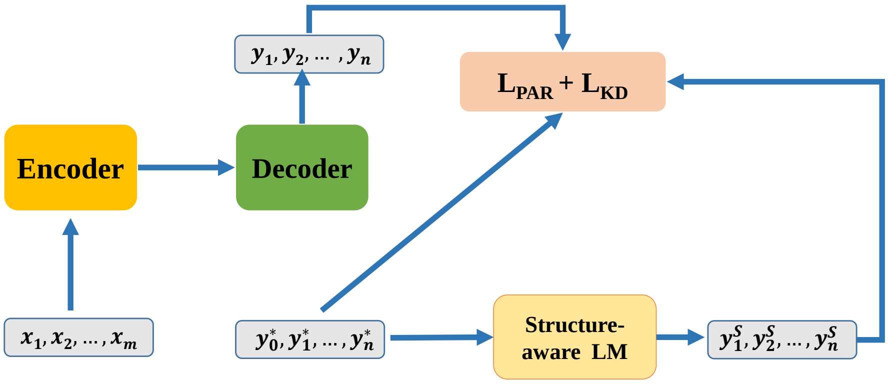



Figure 2: An overview of the proposed model's architecture.

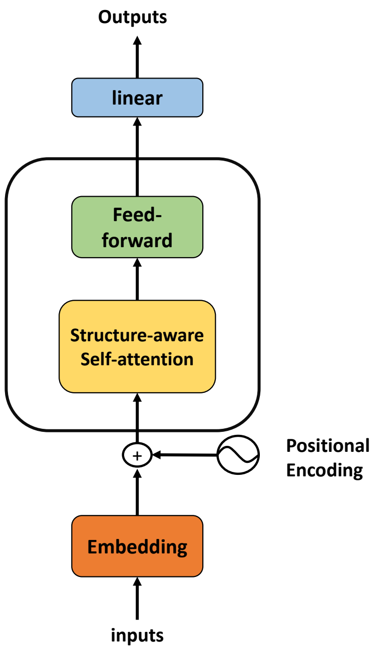



Figure 3: An overview of structure-aware language model's architecture.

The generation of the language model is determined by:

$$P_{LM}(y_t) = p(y_t \mid y_{<t}). \tag{9}$$

In our experiments, the language model is trained based on $\lambda$-calculus expressions and python codes appeared in the training sets of the ATIS, GEO, Django and CoNaLa datasets respectively. The training objective of the language model is to minimize the cross-entropy with target expressions:

$$L_{LM} = -\sum_{i=1}^{N}\sum_{k=1}^{|V|} \mathbf{1}\{y_t = k\} \log P_{LM}(y_t = k), \tag{10}$$

where $N$ is the length of the target sequence, $L_{LM}$ denote the training objective functions for the language model, $P_{LM}$ is computed by Equation (9) respectively.

Given a sequence of preprocessed logic form indexes $\mathbf{y}^* = \{y_0^*, \dots, y_{n-1}^*\}$ obtained from a labeled logical form ($y_0^*$ is the start symbol, $y_n^*$ is the end symbol), the language model produce likelihoods of the target distribution as soft labels, i.e., it generates $\mathbf{y}^S = \{y_1^S, \dots, y_n^S\}$.

### 4.3 Training

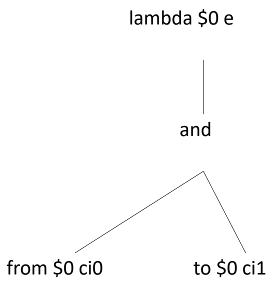
In the training process, we need to combine the loss from the seq2seq model, $L_{PAR}$, and the loss from knowledge distillation, $L_{KD}$.

In specific, to make the seq2seq model learn the knowledge from the language model, we put target sequences into the language model to get estimated probabilities, then we minimize the Kullback-Leibler (KL) divergence between output of the language model and output of the decoder. The loss from knowledge distillation is:

$$L_{KD} = -\sum_{i=t}^{T}\sum_{k=1}^{|V|} KL\big(P_{PAR}(y_t = k) P_{LM}(y_t = k)\big) \tag{11}$$

where $P_{LM}$ denotes the output of the language model computed by by Equation (9) and the function $KL$ computes the KL divergence.

At last, the loss for the entire model is the com-




(a)

| lambda | $0 | e | and | from | $0 | ci0 | to | $0 | ci1 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 3 | 3 | 3 | 3 | 3 | 3 |

(b)

| | lambda | $0 | e | and | from | $0 | ci0 | to | $0 | ci1 |
|---|---|---|---|---|---|---|---|---|---|---|
| lambda | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| e | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| and | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| from | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 0 | 0 | 0 |
| $0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 0 | 0 | 0 |
| ci0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 0 | 0 | 0 |
| to | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 |
| $0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 |
| ci1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 |

(c)

Figure 4: (a): An example of the tree structure of logical form $(\ lambda\ \$0\ e\ (\ and\ (\ from\ \$0\ ci0\ )\ (\ to\ \$0\ ci1\ )\ )\ )$. (b): depth of (a). (c): the structure of (a), which is the input of the structure-aware self-attention.

bination:

$$L = \eta L_{PAR} + (1 - \eta) L_{KD} \tag{12}$$

where $\eta$ is a coefficient between 0 and 1.

## 5 Experiments

In this section, we evaluate our approach on ATIS, GEO, Django and CoNaLa datasets and compare it with other approaches. We also conduct an ablation study to explore the effectiveness of the proposed structure-aware language model.

We first specify details of our implementation including the datasets, the hyperparameters, hardware, and software for training and testing networks. Then we present the experimental results, which show that our model achieves new state-of-the-art performance among various neural semantic parsers on all four datasets.

### 5.1 Datasets

We evaluate our approach on four semantic parsing and code generation benchmarks:

**ATIS** contains natural language questions of a flight dataset paired with a lambda calculus query. We follow the standard train-dev-test split of the datasets in (Zettlemoyer and Collins, 2007), which is 4434/491/448.

**GEO** contains natural language questions about US geography paired with Prolog database queries. We use the corresponding $\lambda$-calculus expressions with the same meaning as in (Kwiatkowski et al., 2011). We follow the standard train-dev-test split of the datasets in (Zettlemoyer and Collins, 2005), which is 600/0/280.

**Django** contains lines of Python source code extracted from the Django framework paried with an NL description. We follow the standard train-dev-test split of the datasets in (Oda et al., 2015) , which is 16000/1000/1805.

**CoNaLa** contains mannully annotated NL questions paired with python solution on STACKOVERFLOW. We follow the standard train-dev-test split of the datasets in (Yin et al., 2018), which is 2379/0/500.

| Model | ATIS | GEO |
|---|---|---|
| ZC07(Zettlemoyer and Collins, 2007) | 84.6 | 86.1 |
| FUBL(Kwiatkowski et al., 2011) | 82.8 | 88.6 |
| KCAZ13(Kwiatkowski et al., 2013) | - | 89.0 |
| **Neural network models** | | |
| Seq2Seq(Dong and Lapata, 2016) | 84.2 | 84.6 |
| Seq2Tree(Dong and Lapata, 2016) | 84.6 | 87.1 |
| JL16(Jia and Liang, 2016) | 83.3 | 89.3 |
| TranX(Yin and Neubig, 2018) | 86.2 | 88.2 |
| Coarse2fine(Dong and Lapata, 2018) | 87.7 | 88.2 |
| Seq2Act(Chen et al., 2018) | 87.7 | 88.2 |
| Graph2Seq(Xu et al., 2018) | 85.5 | 88.9 |
| AdaNSP (Zhang et al., 2019) | 88.6 | 88.9 |
| GNN(Shaw et al., 2019) | 87.1 | 89.3 |
| TreeGen(Sun et al., 2020) | 89.1 | 89.6 |
| PASCAL+CA(Xie et al., 2021) | 90.2 | 90.7 |
| **Ours** | | |
| Baseline | 88.6 | 88.9 |
| + SLM KD fusion | **90.4** | **91.1** |
| - structure-aware | 88.8 | 89.3 |

Table 1: Results on ATIS and GEO datasets

### 5.2 Implementation Details

We use AdaNSP(Zhang et al., 2019), a competitive seq2seq semantic parsing model built on AllenNLP(Wallace et al., 2019), as our base model for

| Model | Django |
|---|---|
| TranX(Yin and Neubig, 2018) | 73.7 |
| Coarse2fine(Dong and Lapata, 2018) | 74.1 |
| TranX2 (Yin and Neubig, 2019) | 77.3±0.4 |
| TranX2+BERT | 79.7±0.42 |
| TAE (Norouzi et al., 2021) | 81.03±0.14 |
| **Ours** | |
| Baseline | 81.03 |
| + SLM KD fusion | **81.83** |
| - structure-aware | 81.16 |

Table 2: Results on Django dataset

| Model | CoNaLa |
|---|---|
| TranX(Yin and Neubig, 2018) | 24.3 |
| EK (Xu et al., 2020) | 27.20 |
| EK+100k (Xu et al., 2020) | 28.14 |
| EK+100K+API (Xu et al., 2020) | 32.26 |
| TAE (Norouzi et al., 2021) | 32.57±0.3 |
| **Ours** | |
| Baseline | 32.57 |
| + SLM KD fusion | **33.10** |
| - structure-aware | 32.62 |

Table 3: Results on CoNaLa dataset

two semantic parsing tasks. The model uses adaptive decoding method that guide the decoder by model uncertainty and automatically uses deeper computations when necessary. The AdaNSP model is not the state-of-the-art model now, but it is based on seq2seq architecture and open-sourced so it is easy to implement our method. We adapt the same hypeparameters as in (Zhang et al., 2019). We use TAE (Norouzi et al., 2021), a seq2seq code generation model as our base model for two code generation tasks. The model exploit a relatively sizeable monolingual corpus of the target programming language to a transformer-based seq2seq model and reach a superior performance.

We trained our model with the hyperparameters listed in Table 5, which was chosen based on the performance of the model on the validation set for ATIS, Django and on the randomly selected training set for GEO, CoNaLa, where the validation set is not provided. For structures of the language model, we set the number of layers 3, positional feed forward dimensions 512, and attention heads 8.We trained the parsing model with the original settings of the baseline system. We trained the language model for 100 epochs respectively, and the entire model for 200 epochs on an Nvidia GeForce RTX 3090 GPU, which takes around 5 hours. Table 4 summarizes numbers of parameters on four datasets for our model.

| Datasets | Numbers of parameters(M) |
|---|---|
| ATIS | 6.68 |
| GEO | 6.62 |
| Django | 9.52 |
| CoNaLa | 9.52 |

Table 4: Numbers of parameters of the SLM for each datasets.

| HYperparameter | Value |
|---|---|
| learning rate | 0.0005 |
| batch size | 256 |
| dropout | 0.1 |
| $\eta$ | 0.8 |
| $k$ | 10 |

Table 5: Hyperparameters.

### 5.3 Evaluation

We use logical form accuracy as the evaluation metric for ATIS and GEO datasets, which is computed with pared trees of the predictions and gold logical forms. The order of the children can be changed within conjunction nodes. We use STree parser code from (Dong and Lapata, 2018) to parse the target lambda expressions and predictions into bracket trees and compare them. We use exact match accuracy as the evaluation metric for Django dataset and corpus-level BLEU for CoNaLa. We report the max results across 5 random seeds.

### 5.4 Results

We compare our method with state-of-the-art semantic parsers on ATIS, GEO, Django and CoNaLa datasets. Table 1- 3 show the results of our model and existing semantic parsers on four datasets. Our model achieves the state-of-the-art performance on four datasets.

We also performed an ablation study by removing the proposed structure-aware self-attention. In specific, we use an original transformer encoder as the language model and integrate it into the parsing model by knowledge distillation. The results show that the model using the structure-aware language model outperforms the one using only original language model.

## 6 Conclusion

In this paper, we present a structure-aware self-attention language model to capture structural information of target representations and propose a knowledge distillation based approach to incorporating the target language model into a seq2seq model. We show that using knowledge distillation from a target language model provides a flexible and scalable way for neural semantic parsers to leverage structural features of target representations. Our method achieves strong results and doesn't need any manually designed rules, sketches or extra corpus.

For future direction, we are interested in exploring other datasets to verify the model's ability for structural data. We will also attempt to integrate grammar rules to this model to have a better performance on semantic parsing tasks.

## References


David Alvarez-Melis and T. Jaakkola. 2017. Tree-structured decoding with doubly-recurrent neural networks. In *Proceedings of the 5th International Conference on Learning Representations (ICLR-17)*.

Ye Bai, Jiangyan Yi, Jianhua Tao, Zhengkun Tian, and Zhengqi Wen. 2019. Learn spelling from teachers: Transferring knowledge from language models to sequence-to-sequence speech recognition. In *Proceedings of the 20th Annual Conference of the International Speech Communication Association (INTERSPEECH-19)*, pages 3795–3799.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse (LAW-13)*, pages 178–186.

Bo Chen, Le Sun, and Xianpei Han. 2018. Sequence-to-action: End-to-end semantic graph generation for semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 766–777, Melbourne, Australia. Association for Computational Linguistics.

D. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, Alexander I. Rudnicky, and E. Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *Proceedings of the workshop on Human Language Technology (HLT-94)*, pages 43–48.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2018. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–742, Melbourne, Australia. Association for Computational Linguistics.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Hueichi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL-16)*, pages 12–22.

Aishwarya Kamath and Rajarshi Das. 2018. A survey on semantic parsing. In *Proceedings of the 1st Conference on Automated Knowledge Base Construction (AKBC-19)*.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526, Copenhagen, Denmark. Association for Computational Linguistics.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP-13)*, pages 1545–1556.

Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. 2011. Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-11)*, pages 1512–1523.

Sajad Norouzi, Keyi Tang, and Yanshuai Cao. 2021. Code generation from natural language with less prior knowledge and more monolingual data. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 776–785, Online. Association for Computational Linguistics.

Yusuke Oda, Hiroyuki Fudaba, Graham Neubig, Hideaki Hata, S. Sakti, T. Toda, and Satoshi Nakamura. 2015. Learning to generate pseudo-code from source code using statistical machine translation (t). *2015 30th IEEE/ACM International Conference on*

*Automated Software Engineering (ASE)*, pages 574–584.

Changhao Shan, Chao Weng, Guangsen Wang, Dan Su, Min Xue Luo, Dong Yu, and Lei Xie. 2019. Component fusion: Learning replaceable language model component for end-to-end speech recognition system. In *Processings of ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-19)*, pages 5361–5635.

Peter Shaw, Philip Massey, Angelica Chen, Francesco Piccinno, and Yasemin Altun. 2019. Generating logical forms from graph representations of text and entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 95–106, Florence, Italy. Association for Computational Linguistics.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2018. Cold fusion: Training seq2seq models together with language models. In *Proceedings of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH-18)*, pages 387–391.

Felix Stahlberg, James Cross, and Veselin Stoyanov. 2018. Simple fusion: Return of the language model. In *Proceedings of the 3rd Conference on Machine Translation: Research Papers (WMT-18)*, pages 204–211.

Zeyu Sun, Qihao Zhu, Lili Mou, Yingfei Xiong, Ge Li, and Lu Zhang. 2019. A grammar-based structural cnn decoder for code generation. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-19)*, pages 7055–7062.

Zeyu Sun, Qihao Zhu, Yingfei Xiong, Yican Sun, Lili Mou, and Lu Zhang. 2020. Treegen: A tree-based transformer architecture for code generation. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-20)*, pages 8984–8991.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems (NIPS-17)*, pages 5998–6008.

Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. 2019. AllenNLP interpret: A framework for explaining predictions of NLP models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 7–12, Hong Kong, China. Association for Computational Linguistics.

Chunyang Xiao, Marc Dymetman, and Claire Gardent. 2016. Sequence-based structured prediction for semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1341–1350, Berlin, Germany. Association for Computational Linguistics.

Defeng Xie, Jianmin Ji, Jiafei Xu, and Ran Ji. 2021. Combining improvements for exploiting dependency trees in neural semantic parsing. In *Pacific Rim International Conference on Artificial Intelligence*, pages 58–72. Springer.

Frank F. Xu, Zhengbao Jiang, Pengcheng Yin, Bogdan Vasilescu, and Graham Neubig. 2020. Incorporating external knowledge through pre-training for natural language to code generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6045–6052, Online. Association for Computational Linguistics.

Kun Xu, Lingfei Wu, Zhiguo Wang, Mo Yu, Liwei Chen, and Vadim Sheinin. 2018. Exploiting rich syntactic information for semantic parsing with graph-to-sequence model. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 918–924, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng Yin, Bowen Deng, Edgar Chen, Bogdan Vasilescu, and Graham Neubig. 2018. Learning to mine aligned code and natural language pairs from stack overflow. *2018 IEEE/ACM 15th International Conference on Mining Software Repositories (MSR)*, pages 476–486.

Pengcheng Yin and Graham Neubig. 2017. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 440–450, Vancouver, Canada. Association for Computational Linguistics.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 7–12, Brussels, Belgium. Association for Computational Linguistics.

Pengcheng Yin and Graham Neubig. 2019. Reranking for neural semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4553–4559, Florence, Italy. Association for Computational Linguistics.

J. Zelle and R. Mooney. 1996. Learning to parse database queries using inductive logic programming.

In *Proceedings of the 13th National Conference on Artificial Intelligence and 8th Innovative Applications of Artificial Intelligence Conference (AAAI-96)*, pages 1050–1055.

Luke Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*, pages 678–687.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: structured classification with probabilistic categorial grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 658–666.

Xiang Zhang, Shizhu He, Kang Liu, and Jun Zhao. 2019. AdaNSP: Uncertainty-driven adaptive decoding in neural semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4265–4270, Florence, Italy. Association for Computational Linguistics.