Unveiling the Black Box of PLMs with Semantic Anchors: Towards Interpretable Neural Semantic Parsing

Lunyiu Nie[1*†], **Jiuding Sun**[1*], **Yanlin Wang**[2‡], **Lun Du**[3], **Shi Han**[3], **Dongmei Zhang**[3], **Lei Hou**[1], **Juanzi Li**[1], **Jidong Zhai**[1]

[1] Department of Computer Science and Technology, Tsinghua University
[2] School of Software Engineering, Sun Yat-sen University
[3] Microsoft Research Asia
{nlx20, sjd22}@mails.tsinghua.edu.cn, wangylin36@mail.sysu.edu.cn,
{lun.du, shihan, dongmeiz}@microsoft.com, {houlei,lijuanzi, zhaijidong}@tsinghua.edu.cn

Abstract

The recent prevalence of pretrained language models (PLMs) has dramatically shifted the paradigm of semantic parsing, where the mapping from natural language utterances to structured logical forms is now formulated as a Seq2Seq task. Despite the promising performance, previous PLM-based approaches often suffer from hallucination problems due to their negligence of the structural information contained in the sentence, which essentially constitutes the key semantics of the logical forms. Furthermore, most works treat PLM as a black box in which the generation process of the target logical form is hidden beneath the decoder modules, which greatly hinders the model's intrinsic interpretability. To address these two issues, we propose to incorporate the current PLMs with a hierarchical decoder network. By taking the first-principle structures as the semantic anchors, we propose two novel intermediate supervision tasks, namely *Semantic Anchor Extraction* and *Semantic Anchor Alignment*, for training the hierarchical decoders and probing the model intermediate representations in a self-adaptive manner alongside the fine-tuning process. We conduct intensive experiments on several semantic parsing benchmarks and demonstrate that our approach can consistently outperform the baselines. More importantly, by analyzing the intermediate representations of the hierarchical decoders, our approach also makes a huge step toward the intrinsic interpretability of PLMs in the domain of semantic parsing.

1 Introduction

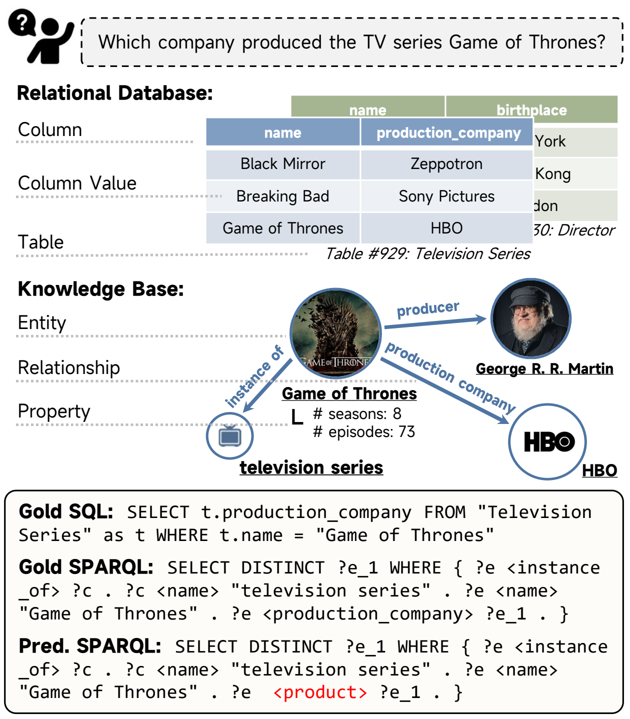
Semantic parsing refers to the task of converting natural language utterances into machine-executable logical forms (Kamath and Das 2019). With the rise of pretrained language models (PLMs) in natural language processing, most recent works in the field formulate semantic parsing as a Seq2Seq task and develop neural semantic parsers on top of the latest PLMs like T5 (Raffel et al. 2020), BART (Lewis et al. 2020), and GPT-3 (Brown et al. 2020), which significantly reduces the manual effort needed in designing compositional grammars (Liang, Jordan, and Klein 2011; Zettlemoyer and Collins 2005). By leveraging the extensive knowledge learned from the pretrain corpus, these PLM-based models exhibit strong performance in comprehending the semantics underlying the source natural language utterance and generating the target logical form that adheres to specific syntactic structures (Shin and Van Durme 2022; Yin et al. 2022).

Despite the promising performance, current PLM-based approaches most regard both input and output as plain text sequences and neglect the structural information contained in the sentences (Yin et al. 2020; Shi et al. 2021), such as the database (DB) or knowledge base (KB) schema that essentially constitutes the key semantics of the target SQL or SPARQL logical forms. As a result, these PLM-based models often suffer from the hallucination issue (Ji et al. 2022) and may generate incorrect logical form structures that are unfaithful to the input utterance (Nicosia, Qu, and Altun 2021; Gupta et al. 2022). For example, as shown in Figure 1, the PLM mistakenly generates a relationship "`product`" in the SPARQL query, which is contradictory to the "*company produced*" mentioned in the natural language.

To prevent the PLMs from generating hallucinated structures, many works propose execution-guided decoding strategies (Wang et al. 2018; Wang, Lapata, and Titov 2021; Ren et al. 2021) and grammar-constrained decoding algorithms (Shin et al. 2021; Scholak, Schucher, and Bahdanau 2021). However, manipulating the decoding process with conditional branches can significantly slow down the model inference (Post and Vilar 2018; Hui et al. 2021). More importantly, in these methods, the DB/KB schema is employed extrinsically as a posteriori correction afterward the model fine-tuning, whereas the inherent ignorance of logical form structures still remains unsolved in the PLMs.

Therefore, another concurrent line of work further

*These authors contributed equally.
†Work done during internship at Microsoft Research Asia.
‡Yanlin Wang is the corresponding author. Work done during the author's employment at Microsoft Research Asia.



Gold SQL: `SELECT t.production_company FROM "Television Series" as t WHERE t.name = "Game of Thrones"`

Gold SPARQL: `SELECT DISTINCT ?e_1 WHERE { ?e <instance_of> ?c . ?c <name> "television series" . ?e <name> "Game of Thrones" . ?e <production_company> ?e_1 . }`

Pred. SPARQL: `SELECT DISTINCT ?e_1 WHERE { ?e <instance_of> ?c . ?c <name> "television series" . ?e <name> "Game of Thrones" . ?e <product> ?e_1 . }`

Figure 1: Example of a natural language utterance and the corresponding SQL & SPARQL logical forms. Specifically, the logical form sequences are composed of schema items that can be aligned to the structure of a database or a knowledge graph. Due to the negligence of these structures, PLM may suffer from hallucination issues and generate unfaithful information, as highlighted in the "Pred. SPARQL".

pretrains the PLMs with structure-augmented objectives (Herzig et al. 2020; Deng et al. 2021). Specifically, these works usually design unsupervised or weakly-supervised objectives for implicitly modeling the database structures with external or synthetic data corpus (Yu et al. 2021b; Shi et al. 2022). Although effective, further pretraining a large PLM can incur substantial costs and extra overheads (Yu et al. 2021a). Besides, these methods also lack transferability since the structural knowledge is latently coupled inside the models and cannot be easily adapted to a novel task domain with a completely distinct database or knowledge base schema (Wu et al. 2021). Thus, how to explicitly address the structural information during the PLM fine-tuning process is still an open question yet to be addressed.

Aside from the above issue, existing neural semantic parsers typically treat PLMs as a black box lacking interpretability. Although some works attempt to probe and explain the latent knowledge within the PLMs using the external modules in a post hoc manner (Liu et al. 2021; Chen et al. 2021b; Stevens and Su 2021a), none of the existing works explicitly addresses the intrinsic interpretability of neural semantic parsers. The intermediate process of logical form generation is completely hidden inside the PLM decoders, where the latent knowledge is hard to probe.

To address these challenges, we propose a novel model architecture with intermediate supervision over a hierarchical decoder network. Inspired by the first principle thinking and its successful application in AMR parsing (Cai and Lam 2019), we define "*semantic anchors*" as the building blocks of a logical form that cannot be further decomposed into more basic structures. For example, in a SQL query, semantic anchors include the tables (relations) and columns (attributes) that constitute the fundamental structure of a relational database (Aho, Beeri, and Ullman 1979; Li and Jagadish 2014); in a SPARQL query, semantic anchors include the entities, relationships, and their respective properties that similarly constitute the backbone of a knowledge base (Angles and Gutierrez 2008; Baeza 2013).

Thereby, the semantic parsing process can now be broken down into the subtasks of extracting the semantic anchors from input utterances and subsequently recombining the identified semantic anchors into the target logical form based on certain formal syntax. We accordingly design two intermediate supervision tasks, namely *Semantic Anchor Extraction* and *Semantic Anchor Alignment*, for explicitly guiding the PLMs to address the structural information alongside the model fine-tuning process. Unlike the previous multi-task learning works that regard PLM as a whole (Radford et al. 2019; Aghajanyan et al. 2021; Xie et al. 2022), we propose a hierarchical decoder architecture that self-adaptively attends to the PLM decoder layers for learning the intermediate supervision objectives. Eventually, this framework can equip the PLMs with intrinsic interpretability where the hidden representations of inner decoders originally concealed inside the PLMs are now unveiled for human analysis and investigation.

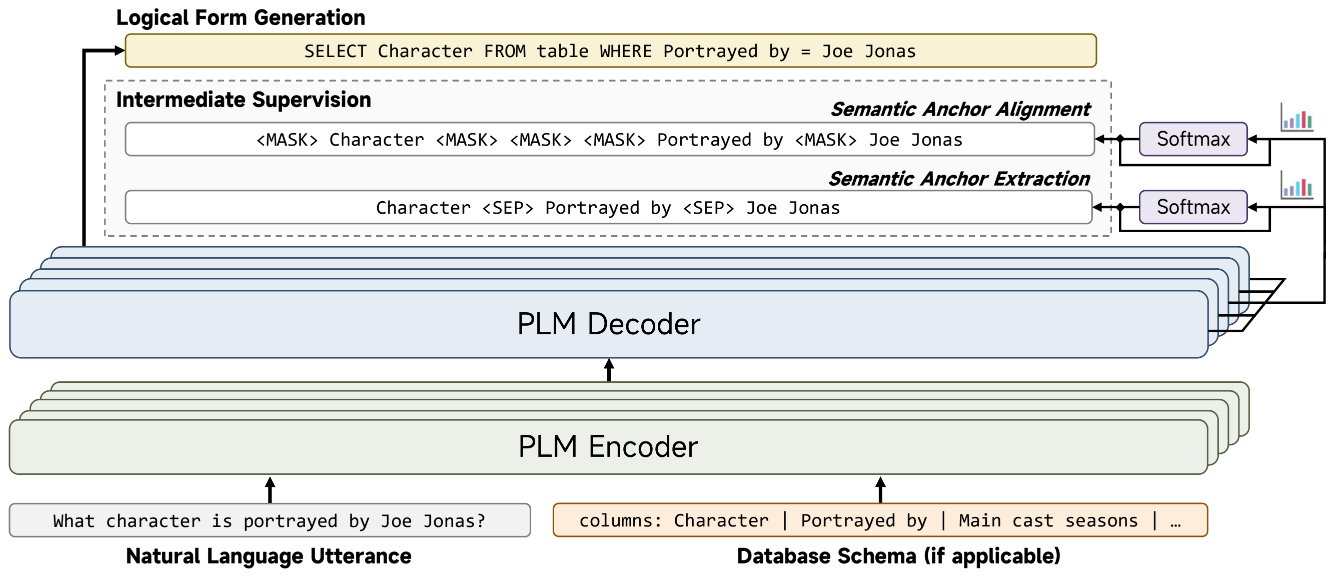
Experimental results show that our proposed framework can consistently improve PLMs' performance on semantic parsing datasets OVERNIGHT, KQA PRO and WIKISQL. By investigating the inner representations of a PLM, our method also provides a novel testbed for interpreting the intermediate process of neural semantic parsing. In summary, our work contributes to the following aspects:

- In this work, we summarize two major issues that hinder the neural semantic parsers: a) negligence of logical form structures, and b) lack of intrinsic interpretability.
- To alleviate the problems, we propose a novel framework with hierarchical decoder and intermediate supervision tasks *Semantic Anchor Extraction* and *Semantic Anchor Alignment* that explicitly highlight the structural information alongside the PLM fine-tuning.
- By investigating the inner layer representations, this is also the first work in the field addressing the intrinsic interpretability of PLM-based semantic parsers.

2 Methodology

Preliminaries

In recent years, pretrained language models (PLMs) like BART (Lewis et al. 2020) and T5 (Raffel et al. 2020) demonstrate strong generalization ability across various Seq2Seq tasks. Within these PLMs, the encoder module first projects the input sequence $\mathbf{x}$ of length m into a sequence of hidden



Logical Form Generation
SELECT Character FROM table WHERE Portrayed by = Joe Jonas
Intermediate Supervision
Semantic Anchor Alignment
<MASK> Character <MASK> <MASK> <MASK> Portrayed by <MASK> Joe Jonas
Semantic Anchor Extraction
Character <SEP> Portrayed by <SEP> Joe Jonas
Softmax
Softmax
PLM Decoder
PLM Encoder
What character is portrayed by Joe Jonas?
columns: Character | Portrayed by | Main cast seasons | …
Natural Language Utterance
Database Schema (if applicable)

Figure 2: Overall framework of the hierarchical decoder that incorporates two intermediate supervision tasks *Semantic Anchor Extraction* and *Semantic Anchor Alignment* for explicitly guiding and probing the PLM alongside the main task fine-tuning.

states $\mathbf{H}_{\mathbb{E}} = \{\mathbf{h}_0, \mathbf{h}_1, ..., \mathbf{h}_m\}$ where each hidden state vector $\mathbf{h}_i$ can be regarded as the contextual embedding of token x_i in the high-dimensional space.

Subsequently, the last encoder hidden states $\mathbf{H}_{\mathbb{E}}^{'}$ is passed to the PLM decoder module consisting of N layer of decoders. Each decoder layer simultaneously takes the previous decoder hidden states for self-attention computation and the last encoder hidden states for cross-attention computation (Vaswani et al. 2017) to produce the new hidden states:

$$\mathbf{H}_{\mathbb{D}}^{i} = \mathbf{Decoder}_i(\mathbf{H}_{\mathbb{D}}^{i-1}, \mathbf{H}_{\mathbb{E}}^{'} \mid \theta_{\mathbb{D}}^{i}), \tag{1}$$

where $\theta_{\mathbb{D}}^{i}$ refers to the i-th decoder layer parameters and $\mathbf{H}_{\mathbb{D}}^{i}$ is the corresponding output hidden states. Eventually, the last decoder hidden states $\mathbf{H}_{\mathbb{D}}^{N}$ are projected into vocabulary-size V-dimensional logits by a linear layer and consequently generates the output tokens once at a time with greedy or beam search decoding.

Therefore, neural semantic parsing can be formally defined as the mapping from a natural language sentence $\mathbf{x} = \{x_1, x_2, ..., x_m\}$ into a logical form sequence $\mathbf{y} = \{y_1, y_2, ..., y_k\}$ by maximizing the conditional probability over PLM parameters θ:

$$p(\mathbf{y}) = \prod_{i=1}^{k} p(y_i|\mathbf{x}, y_1, y_2, ..., y_{i-1}; \theta). \tag{2}$$

.

Semantic Anchor

According to the above formulation, all tokens of a logical form sequence are treated equally by the PLMs, whereas the structural information inside the logical forms is neglected. To analyze the core structures contained in a logical form sequence, we start by giving the formal definitions of knowledge base and relational database.

Knowledge Base A knowledge base (KB), often structured as an RDF graph or property graph, can be defined as a directed graph $G = (N, E)$ where N is a set of nodes (or entities), E is a set of edges (or relationships), $\lambda(N \cup E) \to L$ is a total function that defines the labels of all the nodes and edges, and $\sigma(N \cup E) \to (P, V)$ is a partial function that defines the (property, value) pairs of certain nodes or edges (Angles et al. 2017).

Thereby, for any logical form y querying a knowledge base, we formally define its ***semantic anchors*** as the set of tokens corresponding to the knowledge base schema:

$$\mathcal{S}_{\mathbf{y}|\text{KB}} = \{y_i \in \mathbf{y} | y_i \subset (N \cup E \cup L \cup P \cup V)\}, \tag{3}$$

including the KB entities, relationships, their respective labels, and applicable (property, value) pairs.

Relational Database A relational database (DB) is defined over a database schema D including a set of relational schemas (or tables) $D = \{R_i | 1 \leq i \leq n\}$, where each relational schema further consists a set of attributes schemas (or columns) $R_i = \{A_j^i | 1 \leq j \leq k\}$ (Li and Jagadish 2014)

Thereby, for any logical form y querying a relational database, we formally define its ***semantic anchors*** as the set of tokens aligned to the database schema:

$$\mathcal{S}_{\mathbf{y}|\text{DB}} = \{y_i \in \mathbf{y} | y_i \subset (R \cup A)\}, \tag{4}$$

including the DB table names and column names.

Intermediate Supervision Tasks

Based on the definition of *semantic anchor*, we subsequently design two intermediate supervision tasks by decomposing the semantic parsing process into the subtasks of 1) extracting the *semantic anchors* from the input natural language utterance, then 2) putting the extracted *semantic anchors* into the right positions of a target sequence according to the syntax rule of target formal language.

Semantic Anchor Extraction For the first intermediate supervision task, we enforce the PLMs to extract the *semantic anchors* to explicitly address the logical form structures during the fine-tuning. For each logical form sequence y, we concatenate its *semantic anchors* into a new sequence:

$$\mathbf{y}_{\text{SAE}} = \{\mathcal{S}_{\mathbf{y}}^{1}, \texttt{<SEP>}, \mathcal{S}_{\mathbf{y}}^{2}, \texttt{<SEP>}, ..., \mathcal{S}_{\mathbf{y}}^{s}\}, \quad (5)$$

where `<SEP>` is a special token for separating two distinct *semantic anchors*. A cross-entropy loss is calculated on this extraction supervision and the corresponding tokens at the inner decoder layers.

Semantic Anchor Alignment Thereafter as the second intermediate supervision task, we guide the model to generate the *semantic anchors* with correct relative positions that can be precisely aligned to the final sequence of the target logical form. For each logical form sequence y, we only keep the *semantic anchors* and mask the rest tokens:

$$\mathbf{y}_{\text{SAA}} = \{\texttt{<MASK>}, \texttt{<MASK>}, \mathcal{S}_{\mathbf{y}}^{1}, \texttt{<MASK>}, ..., \mathcal{S}_{\mathbf{y}}^{2}, ...\}, \quad (6)$$

where each *semantic anchor* token $\mathcal{S}_{\mathbf{y}}^{i} \in \mathcal{S}_{\mathbf{y}}$ occurs in the exact relative position as aligned to the target logical form, and the remaining tokens masked by `<MASK>` are ignored during the loss computation.

Hierarchical Decoders

To equip the PLMs with the ability to explicitly address the structural information and improve the intrinsic interpretability of the neural semantic parsers, we want to find a natural way to incorporate the *semantic anchors* during the PLM fine-tuning. For a N-layers decoder module, the inner hidden states are given as $\{\mathbf{H}_{\mathbb{D}}^{i} | 1 \leq i \leq N-1\}$.

For each intermediate supervision task t, we train an independent linear layer f_t and a set of weighting parameters $\{w_t^i | 1 \leq i \leq N-1\}$. Thereby, we can calculate the aggregation of the inner decoder hidden states with a softmax distribution w.r.t. the weighting parameters and a residual connection:

$$\mathbf{H}_{\mathbb{D}}^{t} = \underbrace{\sum_{i=1}^{N-1} \frac{e^{w_t^i}}{\sum_{j=1}^{N-1} e^{w_t^j}} \mathbf{H}_{\mathbb{D}}^{i}}_{\text{learnable weights}} + \underbrace{\sum_{i=1}^{N-1} \frac{1}{N-1} \mathbf{H}_{\mathbb{D}}^{i}}_{\text{residual connection}}. \quad (7)$$

This setup enables the model to attend to the inner decoder layers self-adaptively. The overall representation $\mathbf{H}_{\mathbb{D}}^{s}$ is then mapped into logits in the vocabulary space V by its task-respective linear layer:

$$\boldsymbol{v}_t = f_t(\mathbf{H}_{\mathbb{D}}^{t}), \quad (8)$$

then the probability token distribution can be given by a softmax function, and the cross-entropy loss $\mathcal{L}_s$ of the task can be computed consequently:

$$\begin{aligned} \mathcal{L}_t &= -\sum \sum_{i=1}^{|v|} p(y_{t,i}) \log p(\hat{y}_{t,i}) \\ &= -\sum \sum_{i=1}^{|v|} p(y_{t,i}) \log \frac{exp(\boldsymbol{v}_{t,i})}{\sum_{j=1}^{|v|} exp(\boldsymbol{v}_{t,j})}, \end{aligned} \quad (9)$$

where $y_{t,i}$ is the i-th token of the target sequence y_t for intermediate supervision task t, $\hat{y}_{t,i}$ represents the i-th token of the generated sequence $\hat{y}_t$ for task t, and $\boldsymbol{v}_{t,i}$ stands for the logit score of predicting $\boldsymbol{v}$ as the i-th in the generated sequence.

Self-Adaptive Weighting

Eventually, the overall fine-tuning of a PLM can now be defined as the aggregation of the main task (*i.e.*, logical form generation) and two intermediate supervision tasks:

$$\mathcal{L} = \mathcal{L}_{main} + w_1 \mathcal{L}_{\text{SAE}} + w_2 \mathcal{L}_{\text{SAA}}, \quad (10)$$

where w_1 and w_2 denote the weighting factors for the two intermediate supervision tasks. To minimize the undesired interference between multiple learning objectives, we adopt a loss-balanced task weighting strategy to dynamically adjust the weighting factors throughout the fine-tuning process (Liu, Liang, and Gitter 2019).

Specifically, for each intermediate supervision task t, we compute and store the first batch loss with respect to this task at each epoch, denoted as $\mathcal{L}_{(b_0,t)}$. The loss weighting factor w_t is then dynamically adjusted at each iteration as:

$$w_t = \sqrt{\frac{\mathcal{L}_{(b_j,t)}}{\mathcal{L}_{(b_0,t)}}}, \quad (11)$$

where $\mathcal{L}_{(b_j,t)}$ refers to the real-time loss of task t at batch j.

3 Experiments

Dataset

Overnight OVERNIGHT (Wang, Berant, and Liang 2015) is a popular semantic parsing dataset containing 13,682 examples of natural language question paired with lambda-DCS logical forms across eight data domains so as to explore diverse types of language phenomena. We follow the previous practice (Cao et al. 2019) and randomly sample 20% of the provided training data as a validation set for performance evaluation during the PLM fine-tuning.

KQA Pro KQA PRO (Cao et al. 2022) is a KBQA dataset consisting of 117,790 natural language utterances and corresponded SPARQL queries over the Wikidata knowledge base (Vrandecic and Krötzsch 2014). It widely covers diverse natural language questions with explicitly enhanced linguistic variety and complex query patterns that involve multi-hop reasoning, value comparison, set operations, etc.

WikiSQL WIKISQL (Zhong, Xiong, and Socher 2017) is a classic Text-to-SQL semantic parsing dataset with 80,654 (question, SQL) data pairs grounded on 24,241 Wikipedia tables. Since WikiSQL queries cover only single tables and limited aggregators, previous PLM-based methods have almost achieved upper-bound performance on WikiSQL with the help of further pretraining and execution-guided decoding. Thus in this paper, we only compare to the models without using any additional resources or decoding-aiding techniques for fairness.

	Exec. Acc.
Non-PLM Methods	
SPO (Wang, Berant, and Liang 2015)	58.8
CrossDomain (Su and Yan 2017)	80.6
Seq2Action (Chen, Sun, and Han 2018)	79.0
2-stage DUAL (Cao et al. 2020)	80.1
PLM-based Methods	
T5-base	74.7
Ours (T5-base)	**75.5**
BART-base	80.7
GraphQ IR (BART-base) (Nie et al. 2022)	82.1
Ours (BART-base)	**82.4**
w/o Semantic Anchor Extraction	81.0
w/o Semantic Anchor Alignment	81.5
w/o Hierarchical Decoder	81.2

Table 1: Test accuracies on OVERNIGHT dataset.

	Exec. Acc.
Non-PLM Methods	
EmbedKGQA (Cao et al. 2022)	28.36
RGCN (Cao et al. 2022)	35.07
RNN (Cao et al. 2022)	41.98
PLM-based Methods	
T5-base	83.64
Ours (T5-base)	**84.66**
BART-base (Cao et al. 2022)	89.68
GraphQ IR (BART-base) (Nie et al. 2022)	91.70
Ours (BART-base)	**91.72**
w/o Semantic Anchor Extraction	91.09
w/o Semantic Anchor Alignment	91.12
w/o Hierarchical Decoder	90.94

Table 2: Test accuracies on KQA PRO dataset.

Metric

We use *execution accuracy* as our evaluation metric. It examines whether the generated logical form can be executed by the respective KB or DB engines and return the exact set of results as identical to the ground truth logical forms.

Experimental Settings

We conduct our experiments with $8\times$ NVIDIA Tesla V100 32GB GPUs and the CUDA environment of 10.2. All PLM models used in this work are acquired from the publicly released checkpoints on Huggingface [1]. For BART-base, we fine-tuned the model with a learning rate of $3e-5$ and a warm-up proportion of 0.1. For T5-Base, the learning rate is set to $3e-4$ without warm-up. The batch size is consistently set to 128, and AdamW is used as the optimizer.

Results

Experiment results show that our proposed framework can consistently outperform the baselines on OVERNIGHT, KQA PRO, and WIKISQL datasets, as presented respectively in Table 1, 2, and 3.

[1]https://huggingface.co/models

	Exec. Acc.
Non-PLM Methods	
Seq2SQL (Zhong, Xiong, and Socher 2017)	59.4
Coarse-to-Fine (Dong and Lapata 2018)	78.5
Auxiliary Mapping (Chang et al. 2020)	81.7
PLM-based Methods	
T5-base	84.5
Ours (T5-base)	**85.0**
BART-base	83.6
Ours (BART-base)	**84.8**
w/o Semantic Anchor Extraction	84.3
w/o Semantic Anchor Alignment	84.7
w/o Hierarchical Decoder	84.2
PLM Methods with Additional Resources	
SQLova + EG (Hwang et al. 2019)	86.2
GRAPPA (Yu et al. 2021a)	90.8
SeaD + EG (Xu et al. 2022)	**93.0**

Table 3: Test execution accuracies on WIKISQL dataset. For fairness, we compare our methods with the plain-PLMs that do not use any additional resources (*e.g.*, further pretraining, data augmentation, execution-guided decoding, etc.). The SOTAs are also listed here for readers' information.

Specifically, on both OVERNIGHT and KQA PRO datasets, our framework achieves the new state-of-the-art performance and demonstrates significant accuracy improvement over the PLM baselines. Remarkably, with the aid of intermediate supervision tasks and hierarchical decoder designs, our work even outperforms GraphQ IR (Nie et al. 2022), which requires the laborious implementation of an intermediate representation transpiler.

On WIKISQL, our proposed approach also demonstrates superior execution accuracy over the T5-base and BART-base PLM baselines. In spite of the performance gap compared to the state-of-the-art works on WIKISQL, their works all rely heavily on further pertaining and execution-guided decoding, whereas our approach does not use any external resources other than the datasets themselves.

Ablation Studies

For model ablation, we implement three different settings by removing the respective module:

- **Without Semantic Anchor Extraction** The proposed hierarchical decoder architecture with only the *Semantic Anchor Alignment* task as the intermediate supervision.
- **Without Semantic Anchor Alignment** The proposed hierarchical decoder architecture with only the *Semantic Anchor Extraction* task as the intermediate supervision.
- **Without Hierarchical Decoder** Both *Semantic Anchor Extraction* and *Semantic Anchor Alignment* tasks are performed at the top layer of the PLMs in a multi-task learning setting.

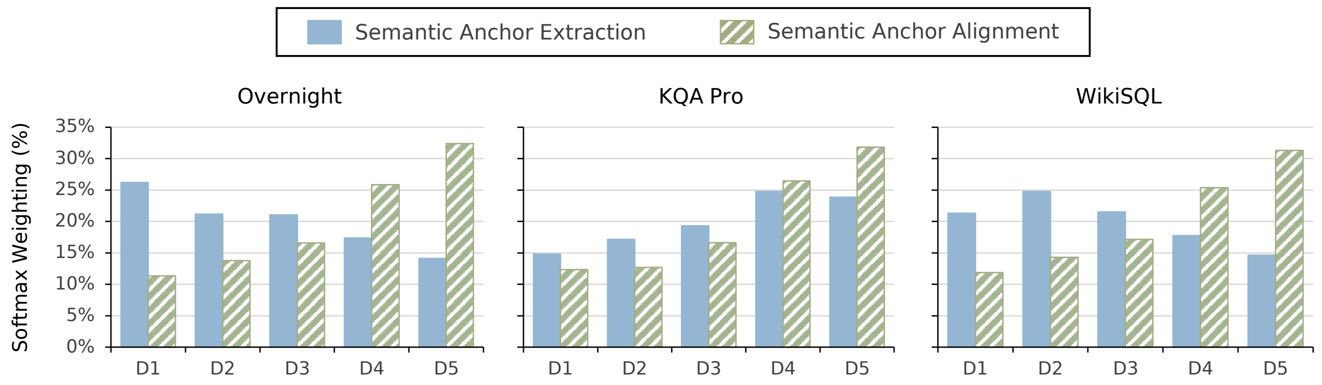


Figure 3: Softmax distribution of the hierarchical decoders over the intermediate supervision tasks of *Semantic Anchor Extraction* and *Semantic Anchor Alignment*. D_i refers to the weighting over the i-th intermediate decoder layer for the specified task.

The ablated experiments demonstrate consistent trends across all of the benchmarks. Models without *Semantic Anchor Extraction* demonstrate a larger drop in performance compared to those without *Semantic Anchor Alignment*. This can be explained as similar to the human cognition process where the extraction of *semantic anchors* is a more fundamental task as the premise of the latter constitution of a whole sequence. On the other hand, models without the hierarchical decoder also degrade significantly, which affirms our hypothesis that guiding the PLMs with supervision over the inner decoder layers can equip the models with improved robustness and intrinsic interpretability.

We notice that the *Alignment* task poses larger impacts on KQA Pro (-0.9%) and Overnight (-0.6%) than on WikiSQL (-0.1%). This can be explained by the relatively shorter logical form length in WikiSQL, which significantly ease the model's learning of the tokens' position alignment.

Hallucination Analysis

To call back the motivation and evaluate whether our proposed framework can help alleviate the hallucination issues in PLM-based semantic parsers, we further compare and analyze the generated logical forms from our method and from the BART-base baseline. We determine hallucination based on whether the generated logical form contains unfaithful or irrelevant schema tokens (Shi et al. 2021; Ji et al. 2022), and the results are shown in Table 4. By explicitly addressing the *semantic anchors* with intermediate supervision, our method can enforce the PLM to generate faithful structures and significantly reduce hallucination. Apart from the quantitative results, we also conduct case analysis and present two examples from the KQA PRO dataset in Figure 4, where the PLM baseline mistakenly generates unfaithful content and our method can precisely output the correct SPARQL query.

Dataset	Baseline	Ours	Difference
OVERNIGHT	294	278	-5.76%
KQA PRO	949	855	-10.99%
WIKISQL	372	334	-11.38%

Table 4: Number of hallucination errors made by the BART-base baseline and our model.

Case #1:

NL Question: Tell me the TV series that has the hash tag of history.

Baseline generated SPARQL: `SELECT DISTINCT ?e WHERE { ?e <pred:instance_of> ?c . ?c <pred:name> "television station" . ?e <hashtag> ?pv . ?pv <pred:value> "history" }`

Ours generated SPARQL: `SELECT DISTINCT ?e WHERE { ?e <pred:instance_of> ?c . ?c <pred:name> "television series" . ?e <hashtag> ?pv . ?pv <pred:value> "history" }`

Case #2:

NL Question: Which film has musician Jimi Hendrix as its subject?

Baseline generated SPARQL: `SELECT DISTINCT ?e WHERE { ?e <pred:instance_of> ?c . ?c <pred:name> "film" . ?e <narrative_location> ?e_1 . ?e_1 <pred:name> "Jimi Hendrix" . ?e_1 <occupation> ?e_2 . ?e_2 <pred:name> "musician" }`

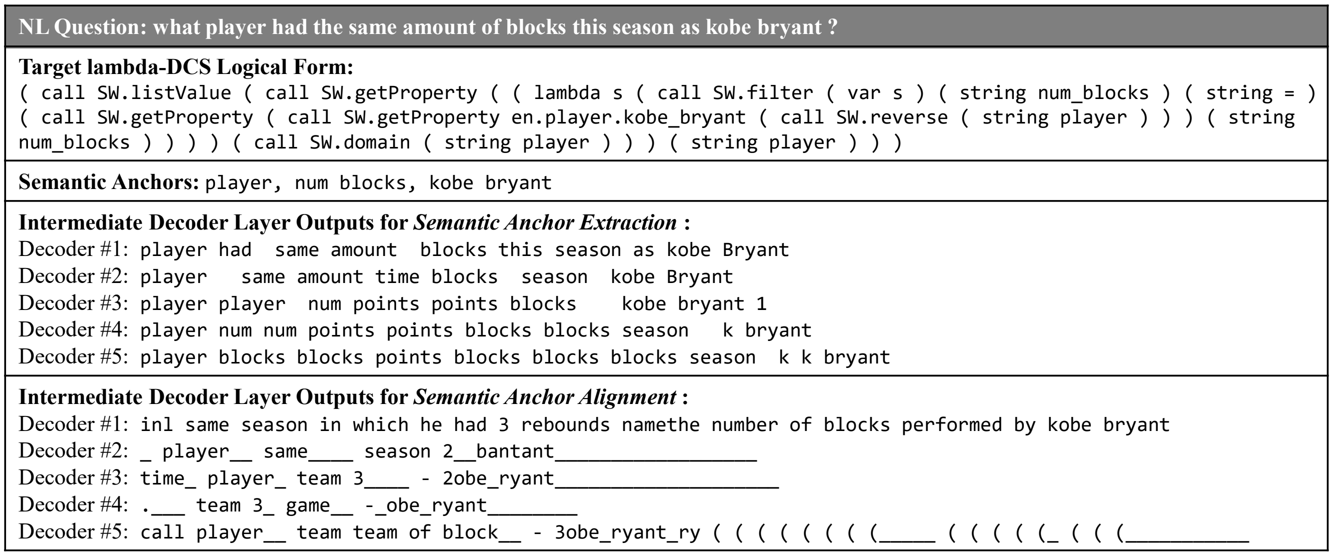
Ours generated SPARQL: `SELECT DISTINCT ?e WHERE { ?e <pred:instance_of> ?c . ?c <pred:name> "film" . ?e <depicts> ?e_1 . ?e_1 <pred:name> "Jimi Hendrix" . ?e_1 <occupation> ?e_2 . ?e_2 <pred:name> "musician" }`

Figure 4: Case analysis of the PLM hallucination issues.

Interpretability Analysis

By performing intermediate supervision over the inner decoder layers together with the main task fine-tuning, this work also provides a novel testbed for probing the latent knowledge hidden inside a large PLM. Specifically, our framework can equip the PLMs with intrinsic interpretability in the following aspects.

Hierarchical Decoder Distribution As aforementioned in Section 2, during the intermediate supervision, the model can self-adaptively attend to the optimal inner decoder layers with weightings adjusted by loss backpropagation. Therefore, by analyzing the weighting distribution over the hierarchical decoders, we can thereby examine the sublayer functionalities of the PLM decoders. As can be observed in



NL Question: what player had the same amount of blocks this season as kobe bryant ?

Target lambda-DCS Logical Form:
```
( call SW.listValue ( call SW.getProperty ( ( lambda s ( call SW.filter ( var s ) ( string num_blocks ) ( string = )
( call SW.getProperty ( call SW.getProperty en.player.kobe_bryant ( call SW.reverse ( string player ) ) ) ( string
num_blocks ) ) ) ) ( call SW.domain ( string player ) ) ) ( string player ) ) )
```

Semantic Anchors: `player, num blocks, kobe bryant`

Intermediate Decoder Layer Outputs for *Semantic Anchor Extraction* :
```
Decoder #1: player had  same amount  blocks this season as kobe Bryant
Decoder #2: player   same amount time blocks  season  kobe Bryant
Decoder #3: player player  num points points blocks    kobe bryant 1
Decoder #4: player num num points points blocks blocks season   k bryant
Decoder #5: player blocks blocks points blocks blocks blocks season  k k bryant
```

Intermediate Decoder Layer Outputs for *Semantic Anchor Alignment* :
```
Decoder #1: inl same season in which he had 3 rebounds namethe number of blocks performed by kobe bryant
Decoder #2: _ player__ same____ season 2__bantant__________________
Decoder #3: time_ player_ team 3____ - 2obe_ryant____________________
Decoder #4: .___ team 3_ game__ -_obe_ryant________
Decoder #5: call player__ team team of block__ - 3obe_ryant_ry ( ( ( ( ( ( ( (_____ ( ( ( ( (_ ( ( (___________
```

Figure 5: BART-base inner decoder layer outputs respectively for the two intermediate supervision tasks.

Figure 3, PLMs tend to perform *Extraction* at the lower decoder layers and *Alignment* at the upper layers, which can be exactly aligned to the order of how humans may process the semantic parsing tasks.

Intermediate Layer Output Analysis More importantly, with our proposed hierarchical decoder, the PLM users are now able to probe the hidden representations of inner decoder layers. Specifically, by converting the layer-wise hidden logits into human-readable outputs, our work can be extended to understand the internal mechanisms behind PLM processing. We present an example from OVERNIGHT dataset with the BART-base inner 5 decoder layer outputs in Figure 5. We conclude that the lower layers of the PLM are more likely to contain information from the input sequence (*e.g.*, the Decoder #1 outputs for both tasks are quite similar to the input natural language question). As the model hidden representations are further processed, the upper decoder layer outputs have moved closer to the target sequence (*e.g.*, the Decoder #5 output for *Semantic Anchor Alignment* already contain some syntax-related tokens like "`call`" and "`(`" in lambda-DCS logical forms). Some irrelevant tokens (*e.g.*, `points`) also exist in the inner layer output, which provides an indication for potential hallucination errors. Overall, these human-understandable inner layer outputs can greatly improve intrinsic interpretability by unveiling the latent representations from PLMs' black box.

4 Related Work

Semantic Parsing

Semantic parsing is the task of converting natural language utterances into either downstream logical forms (Sun et al. 2020; Zhong, Xiong, and Socher 2017; Yu et al. 2018; Shin and Van Durme 2022) or meaning representations (Banarescu et al. 2013). Non-PLM-based methods in this field tend to incorporate representation learning or graph neural networks. Saxena, Tripathi, and Talukdar infuses knowledge representation to facilitate reasoning over the knowledge base (2020). Schlichtkrull et al. models the knowledge base as a graph by using GCN (2018). Recent studies obtain state-of-the-art results on semantic parsing datasets with the capability of existing PLMs (Shin and Van Durme 2022; Chen et al. 2021a) or further pretrained domain-specific PLMs (Yin et al. 2020; Herzig et al. 2020; Yu et al. 2021a).

PLM Interpretability

Most current studies attempt to analyze the hidden representations inside the PLMs with post-hoc methods such as attention analysis (Clark et al. 2019) and counterfactual manipulation (Stevens and Su 2021b). Among them, Yang et al. study the functionalities of each decoder sublayer in Transformer by analyzing the attention map while performing machine translation (2020). Shi et al. take a more proactive approach to supervise the attention with prior knowledge during the training process (2020). Liu et al. explore the grounding capacity of PLMs by erasing the input tokens in sequential order to observe the change of confidence of each concept to be predicted (2021).

5 Conclusion

In this paper, we address the two major issues in PLM-based semantic parsers. We design two intermediate supervision tasks, *Semantic Anchor Extraction* and *Semantic Anchor Alignment*, to guide the PLM fine-tuning through a novel self-adaptive hierarchical decoder architecture. Extensive experiments show our framework can significantly reduce hallucination errors and improve model intrinsic interpretability by probing the inner decoder layers' hidden representations into human-readable outputs. Extensive experiments show our model can consistently outperform the PLM baselines, reduce hallucination errors, and demonstrate improved intrinsic interpretability.

References

Aghajanyan, A.; Gupta, A.; Shrivastava, A.; Chen, X.; Zettlemoyer, L.; and Gupta, S. 2021. Muppet: Massive Multi-task Representations with Pre-Finetuning. In *Proc. of EMNLP*.

Aho, A. V.; Beeri, C.; and Ullman, J. D. 1979. The theory of joins in relational databases. *ACM Transactions on Database Systems (TODS)*.

Angles, R.; Arenas, M.; Barceló, P.; Hogan, A.; Reutter, J. L.; and Vrgoc, D. 2017. Foundations of Modern Query Languages for Graph Databases. *ACM Comput. Surv.*

Angles, R.; and Gutierrez, C. 2008. Survey of graph database models. *ACM Computing Surveys (CSUR)*.

Baeza, P. B. 2013. Querying graph databases. In *Proc. of PODS*.

Banarescu, L.; Bonial, C.; Cai, S.; Georgescu, M.; Griffitt, K.; Hermjakob, U.; Knight, K.; Koehn, P.; Palmer, M.; and Schneider, N. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. In *Proc. of NeurIPS*.

Cai, D.; and Lam, W. 2019. Core Semantic First: A Top-down Approach for AMR Parsing. In *Proc. of EMNLP*.

Cao, R.; Zhu, S.; Liu, C.; Li, J.; and Yu, K. 2019. Semantic Parsing with Dual Learning. In *Proc. of ACL*.

Cao, R.; Zhu, S.; Yang, C.; Liu, C.; Ma, R.; Zhao, Y.; Chen, L.; and Yu, K. 2020. Unsupervised Dual Paraphrasing for Two-stage Semantic Parsing. In *Proc. of ACL*.

Cao, S.; Shi, J.; Pan, L.; Nie, L.; Xiang, Y.; Hou, L.; Li, J.; He, B.; and Zhang, H. 2022. KQA Pro: A Dataset with Explicit Compositional Programs for Complex Question Answering over Knowledge Base. In *Proc. of ACL*.

Chang, S.; Liu, P.; Tang, Y.; Huang, J.; He, X.; and Zhou, B. 2020. Zero-Shot Text-to-SQL Learning with Auxiliary Task. In *Proc. of AAAI*.

Chen, B.; Sun, L.; and Han, X. 2018. Sequence-to-Action: End-to-End Semantic Graph Generation for Semantic Parsing. In *Proc. of ACL*.

Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. d. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; et al. 2021a. Evaluating large language models trained on code. *ArXiv preprint*.

Chen, Z.; Chen, L.; Li, H.; Cao, R.; Ma, D.; Wu, M.; and Yu, K. 2021b. Decoupled Dialogue Modeling and Semantic Parsing for Multi-Turn Text-to-SQL. In *Proc. of ACL Findings*.

Clark, K.; Khandelwal, U.; Levy, O.; and Manning, C. D. 2019. What Does BERT Look at? An Analysis of BERT's Attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

Deng, X.; Awadallah, A. H.; Meek, C.; Polozov, O.; Sun, H.; and Richardson, M. 2021. Structure-Grounded Pretraining for Text-to-SQL. In *Proc. of NAACL*.

Dong, L.; and Lapata, M. 2018. Coarse-to-Fine Decoding for Neural Semantic Parsing. In *Proc. of ACL*.

Gupta, V.; Shrivastava, A.; Sagar, A.; Aghajanyan, A.; and Savenkov, D. 2022. RetroNLU: Retrieval Augmented Task-Oriented Semantic Parsing. In *Proceedings of the 4th Workshop on NLP for Conversational AI*.

Herzig, J.; Nowak, P. K.; Müller, T.; Piccinno, F.; and Eisenschlos, J. 2020. TaPas: Weakly Supervised Table Parsing via Pre-training. In *Proc. of ACL*.

Hui, B.; Shi, X.; Geng, R.; Li, B.; Li, Y.; Sun, J.; and Zhu, X. 2021. Improving text-to-sql with schema dependency learning. *ArXiv preprint*.

Hwang, W.; Yim, J.; Park, S.; and Seo, M. 2019. A Comprehensive Exploration on WikiSQL with Table-Aware Word Contextualization. *ArXiv preprint*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.; Madotto, A.; and Fung, P. 2022. Survey of hallucination in natural language generation. *ArXiv preprint*.

Kamath, A.; and Das, R. 2019. A Survey on Semantic Parsing. In *Proc. of AKBC*.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proc. of ACL*.

Li, F.; and Jagadish, H. V. 2014. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*.

Liang, P.; Jordan, M.; and Klein, D. 2011. Learning Dependency-Based Compositional Semantics. In *Proc. of ACL*.

Liu, Q.; Yang, D.; Zhang, J.; Guo, J.; Zhou, B.; and Lou, J.-G. 2021. Awakening Latent Grounding from Pretrained Language Models for Semantic Parsing. In *Proc. of ACL Findings*.

Liu, S.; Liang, Y.; and Gitter, A. 2019. Loss-Balanced Task Weighting to Reduce Negative Transfer in Multi-Task Learning. In *Proc. of AAAI*.

Nicosia, M.; Qu, Z.; and Altun, Y. 2021. Translate & Fill: Improving Zero-Shot Multilingual Semantic Parsing with Synthetic Data. In *Proc. of EMNLP Findings*.

Nie, L.; Cao, S.; Shi, J.; Tian, Q.; Hou, L.; Li, J.; and Zhai, J. 2022. GraphQ IR: Unifying Semantic Parsing of Graph Query Language with Intermediate Representation. *ArXiv preprint*.

Post, M.; and Vilar, D. 2018. Fast Lexically Constrained Decoding with Dynamic Beam Allocation for Neural Machine Translation. In *Proc. of NAACL*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I.; et al. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*

Ren, H.; Dai, H.; Dai, B.; Chen, X.; Yasunaga, M.; Sun, H.; Schuurmans, D.; Leskovec, J.; and Zhou, D. 2021. LEGO: Latent Execution-Guided Reasoning for Multi-Hop Question Answering on Knowledge Graphs. In *Proc. of ICML*.

Saxena, A.; Tripathi, A.; and Talukdar, P. 2020. Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings. In *Proc. of ACL*.

Schlichtkrull, M. S.; Kipf, T. N.; Bloem, P.; van den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*.

Scholak, T.; Schucher, N.; and Bahdanau, D. 2021. PICARD: Parsing Incrementally for Constrained Auto-Regressive Decoding from Language Models. In *Proc. of EMNLP*.

Shi, P.; Ng, P.; Nan, F.; Zhu, H.; Wang, J.; Jiang, J.; Li, A. H.; Chakravarti, R.; Weidner, D.; Xiang, B.; et al. 2022. Generation-focused Table-based Intermediate Pre-training for Free-form Question Answering. In *Proc. of AAAI*.

Shi, P.; Ng, P.; Wang, Z.; Zhu, H.; Li, A. H.; Wang, J.; dos Santos, C. N.; and Xiang, B. 2021. Learning contextual representations for semantic parsing with generation-augmented pre-training. In *Proc. of AAAI*.

Shi, T.; Zhao, C.; Boyd-Graber, J.; Daumé III, H.; and Lee, L. 2020. On the Potential of Lexico-logical Alignments for Semantic Parsing to SQL Queries. In *Proc. of EMNLP Findings*.

Shin, R.; Lin, C.; Thomson, S.; Chen, C.; Roy, S.; Platanios, E. A.; Pauls, A.; Klein, D.; Eisner, J.; and Van Durme, B. 2021. Constrained Language Models Yield Few-Shot Semantic Parsers. In *Proc. of EMNLP*.

Shin, R.; and Van Durme, B. 2022. Few-Shot Semantic Parsing with Language Models Trained on Code. In *Proc. of NAACL*.

Stevens, S.; and Su, Y. 2021a. An Investigation of Language Model Interpretability via Sentence Editing. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*.

Stevens, S.; and Su, Y. 2021b. An Investigation of Language Model Interpretability via Sentence Editing. In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*.

Su, Y.; and Yan, X. 2017. Cross-domain Semantic Parsing via Paraphrasing. In *Proc. of EMNLP*.

Sun, Y.; Zhang, L.; Cheng, G.; and Qu, Y. 2020. SPARQA: Skeleton-Based Semantic Parsing for Complex Questions over Knowledge Bases. In *Proc. of AAAI*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *Proc. of NeurIPS*.

Vrandecic, D.; and Krötzsch, M. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*.

Wang, B.; Lapata, M.; and Titov, I. 2021. Learning from Executions for Semantic Parsing. In *Proc. of NAACL*.

Wang, C.; Tatwawadi, K.; Brockschmidt, M.; Huang, P.-S.; Mao, Y.; Polozov, O.; and Singh, R. 2018. Robust text-to-sql generation with execution-guided decoding. *ArXiv preprint*.

Wang, Y.; Berant, J.; and Liang, P. 2015. Building a Semantic Parser Overnight. In *Proc. of ACL*.

Wu, Z.; Yang, P.; Yu, P.; Zhu, R.; Han, Y.; Li, Y.; Lian, D.; Zeng, K.; and Zhou, J. 2021. A unified transferable model for ml-enhanced dbms. *ArXiv preprint*.

Xie, T.; Wu, C. H.; Shi, P.; Zhong, R.; Scholak, T.; Yasunaga, M.; Wu, C.-S.; Zhong, M.; Yin, P.; Wang, S. I.; et al. 2022. Unifiedskg: Unifying and multi-tasking structured knowledge grounding with text-to-text language models. *ArXiv preprint*.

Xu, K.; Wang, Y.; Wang, Y.; Wang, Z.; Wen, Z.; and Dong, Y. 2022. SeaD: End-to-end Text-to-SQL Generation with Schema-aware Denoising. In *Proc. of ACL Findings*.

Yang, Y.; Wang, L.; Shi, S.; Tadepalli, P.; Lee, S.; and Tu, Z. 2020. On the Sub-layer Functionalities of Transformer Decoder. In *Proc. of EMNLP Findings*.

Yin, P.; Neubig, G.; Yih, W.-t.; and Riedel, S. 2020. TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data. In *Proc. of ACL*.

Yin, P.; Wieting, J.; Sil, A.; and Neubig, G. 2022. On The Ingredients of an Effective Zero-shot Semantic Parser. In *Proc. of ACL*.

Yu, T.; Wu, C.; Lin, X. V.; Wang, B.; Tan, Y. C.; Yang, X.; Radev, D. R.; Socher, R.; and Xiong, C. 2021a. GraPPa: Grammar-Augmented Pre-Training for Table Semantic Parsing. In *Proc. of ICLR*.

Yu, T.; Zhang, R.; Polozov, A.; Meek, C.; and Awadallah, A. H. 2021b. SCoRe: Pre-Training for Context Representation in Conversational Semantic Parsing. In *Proc. of ICLR*.

Yu, T.; Zhang, R.; Yang, K.; Yasunaga, M.; Wang, D.; Li, Z.; Ma, J.; Li, I.; Yao, Q.; Roman, S.; Zhang, Z.; and Radev, D. 2018. Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL Task. In *Proc. of EMNLP*.

Zettlemoyer, L. S.; and Collins, M. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *UAI '05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, Edinburgh, Scotland, July 26-29, 2005*.

Zhong, V.; Xiong, C.; and Socher, R. 2017. Seq2SQL: Generating Structured Queries from Natural Language using Reinforcement Learning. *ArXiv preprint*.