# Awakening Latent Grounding from Pretrained Language Models for Semantic Parsing

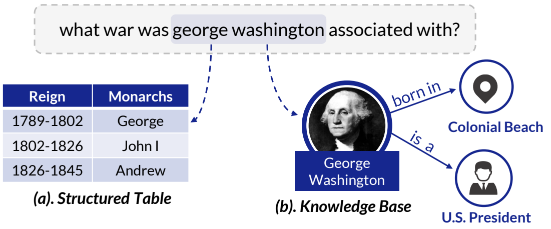
**Qian Liu[†*], Dejian Yang[§], Jiahui Zhang[†*], Jiaqi Guo[◊*], Bin Zhou[†], Jian-Guang Lou[§]**
[†]Beihang University, Beijing, China
[§]Microsoft Research, Beijing, China
[◊]Xi'an Jiaotong University, Xi'an, China
[†]{qian.liu, 17231043, zhoubin}@buaa.edu.cn; [◊]jasperguo2013@stu.xjtu.edu.cn
[§]{dejian.yang, jlou}@microsoft.com

## Abstract

Recent years pretrained language models (PLMs) hit a success on several downstream tasks, showing their power on modeling language. To better understand and leverage what PLMs have learned, several techniques have emerged to explore syntactic structures entailed by PLMs. However, few efforts have been made to explore grounding capabilities of PLMs, which are also essential. In this paper, we highlight the ability of PLMs to discover which token should be grounded to which concept, if combined with our proposed erasing-then-awakening approach. Empirical studies on four datasets demonstrate that our approach can awaken latent grounding which is understandable to human experts, even if it is not exposed to such labels during training. More importantly, our approach shows great potential to benefit downstream semantic parsing models. Taking text-to-SQL as a case study, we successfully couple our approach with two off-the-shelf parsers, obtaining an absolute improvement of up to 9.8%.

## 1 Introduction

Recent breakthroughs of **P**retrained **L**anguage **M**odels (**PLMs**) such as BERT (Devlin et al., 2019) and GPT3 (Brown et al., 2020) have demonstrated the effectiveness of self-supervised learning for a range of downstream tasks. Without being guided by structural information in training, PLMs show the potential for learning implicit syntactic structures and language semantic, which can be transferred to other tasks. To better understand and leverage what PLMs have learned, several work has emerged to probe or induce syntactic structures from PLMs. According to prior studies (Rogers et al., 2020), most existing work focuses on syntactic structures such as part of speech (Liu et al., 2019), constituency tree (Wu et al., 2020) and dependency tree (Hewitt and Manning, 2019; Jawahar et al., 2019), paying much less attention on language semantics (Tenney et al., 2019). However, as well known, semantic information is essential for high-level tasks like machine reading comprehension (Wang and Jiang, 2019).



| Reign | Monarchs |
|---|---|
| 1789-1802 | George |
| 1802-1826 | John I |
| 1826-1845 | Andrew |

Figure 1: Typical scenarios for grounding, here the linguistic tokens "george washington" can be grounded into different real-world concepts.

Regarding to language semantics, an important branch is grounding, which is overlooked by most previous work. Broadly speaking, grounding means "*connecting linguistic symbols to real-world perception or actions*" (Roy, 2005). It is generally thought to be important for a variety of tasks, such as video descriptions (Zhou et al., 2019), visual question answering (Zhu et al., 2016) and semantic parsing (Guo et al., 2019). In this paper, we focus on single-modal scenarios, where grounding refers more specifically to mapping linguistic tokens into a real-world concept described in natural language. As shown in Figure 1, "george washington" can be grounded into either a cell value in a structured table, or an entity in knowledge bases.

In single-modal scenarios, grounding is especially important for semantic parsing, the task of translating a natural language sentence into its corresponding executable logic form. For earlier work, grounding is essential since earlier work almost conceptualized semantic parsing as grounding an

[*]Work done during an internship at Microsoft Research. The first three authors contributed equally.

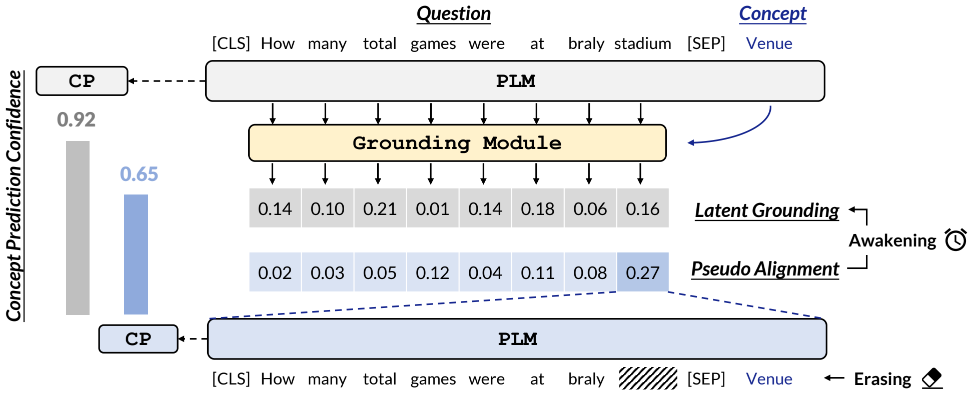


Figure 2: The illustration of ETA, which consists of a PLM module, a **C**oncept **P**rediction (CP) module and a grounding module. Two models (gray and blue) are drawn here for illustration purposes, and they are indeed the same. The model training involves three steps: (1) The concept prediction module is trained to predict the confidence of any concept occurring in a given question (**Left**). (2) The erasing mechanism erases tokens in the question sequentially, feeds them into CP, and obtains the confidence differences (e.g., $0.92 - 0.65 = 0.27$) as the pseudo alignment. Here we only demonstrate the process related to "stadium" (**Bottom Right**). (3) The pseudo alignment is employed to awaken the latent grounding, i.e., to supervise the grounding module (**Top Right**). We show only one concept "Venue" for the sake of brevity, which in practice is a sequence of concepts.

utterance to a task-specific meaning representation (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Liang et al., 2013; Cheng et al., 2017). As for modern approaches based on the encoder-decoder architecture, grounding also plays an important role and considerable work has demonstrated the positive effect of it (Guo et al., 2019; Dong et al., 2019; Liu et al., 2020a; Wang et al., 2020b; Chen et al., 2020). Despite its success, existing grounding methods mainly relied on heavy manual efforts like high-quality lexicons (Reddy et al., 2016) or ad-hoc heuristic rules like n-gram matching (Guo et al., 2019), suffering from poor flexibility. To explore more flexible methods, researchers recently tried a data-driven way: they collected grounding annotations as supervision to train grounding models (Li et al., 2020a; Lei et al., 2020; Shi et al., 2020). However, this modeling flexibility in their approaches requires expensive annotations of grounding, which most of the time are not available.

To alleviate the above issues, we present a novel approach **E**rasing-**t**hen-**A**wakening (ETA)[1]. It is inspired by recent advances in interpretable machine learning (Samek et al., 2017), where the importance of individual pixels can be quantified with respect to the classification decision. Similarly, our approach firstly quantifies the contribution of each word with respect to each concept, by erasing it and probing the variation of concept prediction decisions (elaborated later). Then it employs these contributions as pseudo labels to awaken latent grounding from PLMs. In contrast to prior work, our approach only needs supervision of concept prediction, which can be easily derived by downstream tasks (e.g., text-to-SQL) instead of full grounding supervision. Empirical studies on four datasets demonstrate that our approach can awaken latent grounding which is understandable to human experts. It is highly non-trivial because our approach is not exposed to any human-annotated grounding label in training. More importantly, we find that the grounding can be easily coupled with downstream models to boost their performance, and the absolute improvement is up to $9.8\%$. In summarization, our contribution is as three-fold:

1. To the best of our knowledge, we are the first one to highlight and demonstrate the possibility of awakening latent grounding from PLMs.

2. We propose a novel weakly supervised approach erasing-then-awakening, to awaken latent grounding from PLMs. Empirical studies on four datasets demonstrate that our approach can awaken latent grounding which is understandable to human experts.

3. Taking text-to-SQL as a case study, we successfully couple our approach with two off-the-shelf parsers. Experimental results on two benchmarks show the effectiveness of our approach on boosting downstream performance.

[1] Our code is available at https://github.com/microsoft/ContextualSP

## 2 Method: Erasing-then-Awakening

In the task of grounding, we are given a question $\mathbf{x} = \langle x_1, \cdots, x_N \rangle$ and a concept set $\mathcal{C} = \{c_1, \cdots, c_K\}$, where each concept consists of several tokens. The goal of grounding is to find out tokens (also known as mentions) in $\mathbf{x}$ which are relevant to concepts in $\mathcal{C}$. Generally, the grounding procedure learns to create a $N \times K$ matrix, which we call *latent grounding*. In some cases, a set of pairs is needed, of which each one explicitly shows a token and a concept is grounded. We call this kind of pairs as *grounding pairs* below.

As illustrated in Figure 2, our model consists of a PLM module, a CP module and a grounding module. In this section, we first present the training procedure of ETA, which at a high-level involves three steps: (1) Train an auxiliary concept prediction module. (2) Erase tokens in a question to obtain the concept prediction confidence differences as pseudo alignment. (3) Awaken latent grounding from PLMs by applying pseudo alignment as supervision. Then we introduce the procedure to produce grounding pairs in inference.

### 2.1 Training a Concept Prediction Module

Given $\mathbf{x}$ and $\mathcal{C}$, the goal of the concept prediction module is to identify if each concept $c_k \in \mathcal{C}$ is mentioned or not in the question $\mathbf{x}$. Although it does not seem to be directly related to grounding, it is a pre-requisite for the erasing mechanism, which will be elaborated later. As for $c_k$'s supervision $l_k \in \{0, 1\}$, it is the weak supervision ETA relies on, and can be readily obtained through downstream task signals. Taking text-to-SQL as an illustration, each database schema (i.e., table, column and cell value) in an annotated SQL can be considered as mentioned in the question ($l_k = 1$), with others as negative examples ($l_k = 0$).

Once the supervision is prepared, the CP module is trained to conduct binary classification over the representation of each concept. As done in previous work (Hwang et al., 2019), we first concatenate the question and all concepts into a sequence as input to the PLM module. As illustrated in Figure 2, the input sequence starts with `[CLS]`, with the question and each concept being separated by `[SEP]`. Then, the sequence is fed into the PLM module to produce deep contextual representations over each position. Denoting $\langle \mathbf{q}_1, \mathbf{q}_2, ..., \mathbf{q}_N \rangle$ and $\langle \mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_K \rangle$ as the token representations and concept representations, they can be obtained by:

$$\{\mathbf{q}_n\}_{n=1}^{N}, \{\mathbf{e}_k\}_{k=1}^{K} = \text{PLM}\big(\texttt{[CLS]}, \mathbf{x}, \{\texttt{[SEP]}, c_k\}_{k=1}^{K}\big), \quad (1)$$

where $\mathbf{q}_n$ and $\mathbf{e}_k$ correspond to the representations at the position of $n$-th question token and the first token in $c_k$ respectively. Finally, each concept representation $\mathbf{e}_k$ is passed to a classifier to predict if it is mentioned in $\mathbf{x}$ as:

$$p_k = \texttt{Sigmoid}(\mathbf{W}_l \, \mathbf{e}_k), \quad (2)$$

where $\mathbf{W}_l$ is a learnable parameter. $p_k$ is the probability of $c_k$ mentioned in the question, which is referred to by *concept prediction confidence* below.

### 2.2 Erasing Question Tokens

Once the concept prediction module is converged, we apply an erasing mechanism to assist in the following awakening phase. It follows a similar idea from the interpretable document classification (Arras et al., 2016), where a word is considered important for the document classification if removing it and classifying the modified document results in a strong decrease of the classification score. In our case, a token is considered highly relevant to certain concepts if there is a large drop in these concept prediction confidences after erasing the token. Therefore, we need the above mentioned concept prediction module to provide a reasonable concept prediction confidence.

Concretely, as shown in Figure 2, the erasing mechanism erases the input sequentially, and feeds each erased input into the PLM module and the subsequent CP module. For example, with $x_n$ being substituted by a special token `[UNK]`, we can obtain an erased input as $\texttt{[CLS]}, x_1, \cdots, x_{n-1}, \texttt{[UNK]}, x_{n+1}, \cdots, c_K$. Denoting $\hat{p}_{n,k}$ the concept prediction confidence for $c_k$ after erasing $x_n$, we believe the difference between $\hat{p}_{n,k}$ and $p_k$ reveals $c_k$'s relevance to $x_n$ from a PLM's view. The confidence difference $\Delta_{n,k}$ can be obtained by $\Delta_{n,k} = l_k \cdot \max(0, p_k - \hat{p}_{n,k})$. Repeating the above procedure on the input question sequentially, $\Delta \in \mathbb{R}^{N \times K}$ is filled completely.

### 2.3 Awakening Latent Grounding

As mentioned above, we believe $\Delta$ reflects the relevance between each token and each concept from a PLM's view. Therefore, we could directly use $\Delta$ as ETA's output. However, according to our preliminary study, the method performs poorly and cannot produce high-quality alignment[2]. Different from directly using $\Delta$, we employ it to "awaken" the latent grounding. To be specific, we introduce a grounding module upon representations of the PLM module and train it using $\Delta$ as pseudo labels (i.e., pseudo alignment). The grounding module first obtains grounding scores $g_{n,k}$ between each question token $x_n$ and each concept $c_k$ based on their deep contextual representations $\mathbf{q}_n$ and $\mathbf{e}_k$ as:

$$g_{n,k} = \frac{\mathbf{W}_e \mathbf{e}_k \cdot (\mathbf{W}_q \mathbf{q}_n)^T}{\sqrt{d}}, \tag{3}$$

where $\mathbf{W}_e, \mathbf{W}_q$ are learnable parameters and $d$ is the dimension of $\mathbf{e}_k$. Then it normalizes the grounding scores into latent grounding $\boldsymbol{\alpha}$ as:

$$\boldsymbol{\alpha}_{n,k} = \frac{\exp(g_{n,k})}{\sum_i \exp(g_{i,k})}. \tag{4}$$

Finally, the grounding module is trained to maximize the likelihood with $\Delta$ as the weight:

$$\sum_n \sum_k \Delta_{n,k} \cdot \log \boldsymbol{\alpha}_{n,k}. \tag{5}$$

### 2.4 Producing Grounding Pair

Repeating erasing and awakening iteratively for epochs until the grounding module converges, we can readily produce grounding pairs. Formally, we aim to obtain a set of pairs, where each pair $\langle x_n, c_k \rangle$ indicates that $x_n$ is grounded to $c_k$. Noticing $c_k$ may contain several tokens, we keep all probabilities in $\boldsymbol{\alpha}_{\cdot,k}$ which exceeds $\tau/|c_k|$, where $\tau$ is a threshold and $|c_k|$ is the number of tokens in $c_k$. Also, taking into account that $x_n$ should be grounded to only one concept, we keep only the highest probability over $\boldsymbol{\alpha}_{n,\cdot}$. Finally, for each pair $\langle x_n, c_k \rangle$, it is thought to be a grounding pair if $\boldsymbol{\alpha}_{n,k}$ is kept and $p_k \geq 0.5$, otherwise it is not.

## 3 Experiments

In this section, we conduct experiments to evaluate if the latent grounding awakened by ETA is understandable to human experts. Here we accomplish the evaluation by comparing the grounding pairs produced by ETA with human annotations.

### 3.1 Experimental Setup

**Datasets** We select two representative grounding tasks where human annotations are available: *schema linking* and *entity linking*. Schema linking is to ground questions into database schemas, while entity linking is to ground questions into entities of knowledge bases. For schema linking, we select SPIDER-L (Lei et al., 2020) and SQUALL (Shi et al., 2020) as our evaluation benchmarks. As mentioned in §2.1, the supervision for our model is obtained from SQL queries. As for entity linking, we select WebQSP$_{\text{EL}}$and GraphQ$_{\text{EL}}$(Sorokin and Gurevych, 2018). The supervision for our model is obtained from SPARQL queries in a similar way.

**Evaluation** For schema linking, as done in previous work (Lei et al., 2020), we report the micro-average precision, recall and F1-score for both columns (Col$_P$, Col$_R$, Col$_F$) and tables (Tab$_P$, Tab$_R$, Tab$_F$). For entity linking, we report the weak matching precision, recall and F1-score for entities (Ent$_P$, Ent$_R$, Ent$_F$). The weak matching metric is a commonly used metric in previous work (Sorokin and Gurevych, 2018), which considers a prediction as correct whenever the correct entity is identified and the predicted mention boundary overlaps with the ground truth boundary. More details can be seen in §A.

**Baselines** For schema linking, we consider four strong baselines. (1) **N-gram Matching** enumerates all n-gram ($n \leq 5$) phrases in a natural language question, and links them to database schemas by fuzzy string matching. (2) **SIM** computes the dot product similarity between each question token and schema using their PLM representations without fine-tuning, to explore grounding capacities of unawakened PLMs. (3) **CONTRAST** learns by comparing the aggregated grounding scores of mentioned schemas with unmentioned ones in a contrastive learning style, as done in Liu et al. (2020b). Concretely, in training, CONTRAST is trained to accomplish the same concept prediction task as our approach. With a similar architecture to the Receiver used in Liu et al. (2020b), it first computes the similarity score between each token and each concept, and then uses max pooling to aggregate the similarity scores of a concept over an utterance into a concept prediction score. Finally, a margin-based loss is used to encourage the baseline to give higher concept prediction scores on mentioned concepts than unmentioned concepts. (4) **SLSQL$_{\text{L}}$ & ALIGN$_{\text{L}}$**. SLSQL$_{\text{L}}$ (ALIGN$_{\text{L}}$) is a learnable schema linking module[3] proposed in

[2] More experimental results can be found in §3.3.

[3] SLSQL and ALIGN use multi-task learning to simultaneously learn schema linking and SQL generation.

| Model | SPIDER-L | | | | | | SQUALL | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\text{Col}_P$ | $\text{Col}_R$ | $\text{Col}_F$ | $\text{Tab}_P$ | $\text{Tab}_R$ | $\text{Tab}_F$ | $\text{Col}_P$ | $\text{Col}_R$ | $\text{Col}_F$ |
| N-gram Matching | 61.4 | 69.1 | 65.1 | 78.2 | 69.6 | 73.6 | 71.6 | 50.8 | 59.4 |
| SIM + BERT | 16.6 | 8.0 | 10.8 | 8.5 | 11.6 | 9.8 | 13.9 | 18.0 | 15.7 |
| CONTRAST + BERT | 83.7 | 68.4 | 75.3 | **84.0** | 76.9 | 80.3 | 47.9 | 31.2 | 37.8 |
| ETA + BERT | **86.1** | **79.3** | **82.5** | 81.1 | **85.3** | **83.1** | **77.3** | **62.4** | **69.0** |
| $\text{SLSQL}_\text{L}$ + $\text{BERT}^\heartsuit$ (Lei et al., 2020) | 82.6 | 82.0 | 82.3 | 80.6 | 84.0 | 82.2 | – | – | – |
| $\text{ALIGN}_\text{L}$ + $\text{BERT}^\heartsuit$ (Shi et al., 2020) | – | – | – | – | – | – | 79.2 | 72.8 | 75.8 |

Table 1: Experimental results on schema linking dev sets. $\heartsuit$ means the model uses schema linking supervision, while other learnable models use weak supervision. +BERT means using BERT as encoder, the same for Table 2.

| Model | $\text{WebQSP}_\text{EL}$ | | | $\text{GraphQ}_\text{EL}$(zero-shot) | | |
|---|---|---|---|---|---|---|
| | $\text{Ent}_P$ | $\text{Ent}_R$ | $\text{Ent}_F$ | $\text{Ent}_P$ | $\text{Ent}_R$ | $\text{Ent}_F$ |
| Heuristic (Sorokin and Gurevych, 2018) | 30.2 | 60.8 | 40.4 | - | - | - |
| ETA + BERT | **76.6** | **72.5** | **74.5** | **43.1** | **42.1** | **42.7** |
| $\text{VCG}^\heartsuit$ (Sorokin and Gurevych, 2018) | 82.4 | 68.3 | 74.7 | 54.1 | 30.6 | 39.0 |
| ELQ + $\text{BERT}^\heartsuit$ (Li et al., 2020a) | 90.0 | 85.0 | 87.4 | 60.1 | 57.2 | 58.6 |

Table 2: Experimental results on entity linking test sets. $\heartsuit$ means the model uses entity linking supervision from $\text{WebQSP}_\text{EL}$, while ETA uses the weak supervision derived from WebQSP. Following previous work (Sorokin and Gurevych, 2018), we use $\text{GraphQ}_\text{EL}$only in the evaluation phase to test the generalization ability of our model.

SLSQL (ALIGN). Unlike our method, these two methods are trained with the full schema linking supervision. Please refer to Shi et al. (2020) and Lei et al. (2020) for more details. Notably, for baselines which require a threshold, we tuned their thresholds based on dev sets for fair comparison.

For entity linking, we compare ETA with three powerful methods. (1) **Heuristic** picks the most frequent entity among the candidates found by string matching over Wikidata. (2) **VCG** (Sorokin and Gurevych, 2018) aggregates and mixes contexts of different granularities to perform entity linking. (3) **ELQ** (Li et al., 2020a) uses a bi-encoder to perform entity linking in one pass, achieving state-of-the-art performance on $\text{WebQSP}_\text{EL}$and $\text{GraphQ}_\text{EL}$. VCG and ELQ utilize entity linking supervision in training, while ETA does not.

**Implementation** For schema linking we follow the procedure in §2.4 to produce grounding pairs to evaluate, while for entity linking we further merge adjacent grounding pairs to produce span-level grounding pairs. We implement ETA in Pytorch (Paszke et al., 2019). With respect to PLMs in experiments, we use the uncased BERT-base (BERT)[4] and BERT-large ($\text{BERT}_\text{L}$) from Transformers library (Wolf et al., 2020). As for the optimizer, we employ AdamW (Loshchilov and Hutter, 2019). More details (e.g., learning rate) of each experiment can be found in §C.1.

### 3.2 Experimental Results

Table 1 shows the experimental results on the schema linking task. As shown, our method outperforms all weakly supervised methods and heuristic-based methods by a large margin. For example, on SPIDER-L, ETA + BERT achieves an absolute improvement of 7.2% $\text{Col}_F$ and 2.8% $\text{Tab}_F$ over the best baseline CONTRAST. The same conclusion can be drawn from the experimental results on the entity linking task shown in Table 2. For instance, ETA + BERT can obtain a high $\text{Ent}_F$ up to 74.5% on $\text{WebQSP}_\text{EL}$, which is a satisfying performance for downstream tasks. All results above demonstrate the superiority of our approach on awakening latent grounding from PLMs. With respect to the reason that PLMs work well on both schema linking and entity linking, it may be because both schema linking and entity linking require text-based semantic matching (e.g., synonyms), which PLMs excel at.

Furthermore, it is very surprising that although not trained under fine-grained grounding supervision, our model is comparable with or slightly worse than the fully supervised models across

[4] Our approach is theoretically applicable to different PLMs. In this paper, we chose BERT as a representative and we leave exploration of different PLMs for future work.

| Error Type | Example Error |
|---|---|
| **Missed Grounding (43.1%)** | How many **points** did arnaud demare receive?<br>`GOLD:` **points**→ “`UCI world tour points`” `PRED:` |
| **Technically Correct (21.0%)** | Total **population** of millbrook first **nation**?<br>`GOLD:` **population**→ “`Population`”<br>`PRED:` **population**→ “`Population`”; **nation**→ “`Community`” |
| **Partially Correct (15.8%)** | Who was **the first winning captain**?<br>`GOLD:` **the first**→ “`Year`”; **winning captain**→ “`Winning Captain`”<br>`PRED:` **first**→ “`Year`”; **winning captain**→ “`Winning Captain`” |
| **Wrong Grounding (10.1%)** | Were the **matinee** and evening performances held **earlier** than the 8th anniversary?<br>`GOLD:` **earlier**→ “`Date`”<br>`PRED:` **matinee**→ “`Performance`”; **earlier**→ “`Date`” |

Table 3: Four main error types made by ETA along with their proportions on SQUALL dataset.

datasets. For instance, on SPIDER-L, our model exceeds the fully supervised baseline $\text{SLSQL}_\text{L}$ by 0.9 points on $\text{Tab}_F$. On SQUALL, our model holds a slightly worse performance than the fully supervised baseline $\text{ALIGN}_\text{L}$. It is highly nontrivial since CONTRAST, the best weakly supervised baseline on SPIDER-L, is far from the fully supervised model on SQUALL, while our model has only a small drop. Besides, on $\text{WebQSP}_\text{EL}$and $\text{GraphQ}_\text{EL}$, although our model is inferior to the state-of-the-art model ELQ, it also achieves a comparable performance with the fully supervised baseline VCG. These results provide strong evidence that PLMs do have very good grounding capabilities, and our approach can awaken them from PLMs.

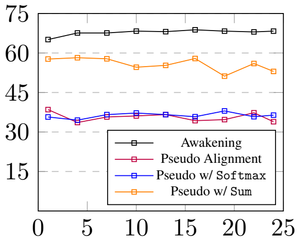


Figure 3: $\text{Col}_F$ score on the dev set of SQUALL at different training epochs. “Pseudo w/ `Softmax`” means normalizing pseudo alignment with `Softmax`, while “Pseudo w/ `Sum`” means normalizing through dividing each number by the sum of them.

### 3.3 Model Analysis

In this section, we try to answer four interesting research questions via a thorough analysis: **RQ1.** Does the grounding capability come mainly from the PLM? **RQ2.** Is the awakening phase necessary? **RQ3.** Do larger PLMs have better grounding capabilities? **RQ4.** What are the remaining errors?

**RQ1** There is a long term debate in literature about if knowledge is primarily learned by PLMs, when extra parameters are employed in analysis (Hewitt and Liang, 2019). Similarly, since our approach depends on extra modules (e.g., grounding module), it faces the same dilemma: how can we know whether the latent grounding is learnt from PLMs or extra modules? Therefore, we apply our approach to a randomly initialized Transformer encoder (Vaswani et al., 2017), to probe the grounding capability of a model that has not been pretrained. To make it comparable, the encoder has the same architecture as BERT. However, it only gets a 40% $\text{Col}_F$ on SQUALL, not even as good as the N-gram baseline. Considering it contains the same extra modules as ETA + BERT, the huge gap between it and ETA + BERT supports the opinion that the latent grounding is mainly learnt from PLMs. Meanwhile, one concern shared by our reviewers is the risk of supervision exposure during training of the concept prediction module. In other words, our approach may “steal” some supervision in the concept prediction module to achieve good performance on grounding. However, the above experiment demonstrates that a non-pretrained model is far from strong grounding capability even with the same concept prediction module. We hope the finding will alleviate the concern.

**RQ2** As mentioned in §2.3, the pseudo alignment $\Delta$ can also be employed as the model prediction. Therefore, we conduct experiments to verify if our proposed awakening phase is necessary. As shown in Figure 3, even with various normalization methods (e.g., `Softmax`), $\Delta$ does not produce satisfactory alignment. In contrast, our model consistently performs well. To investigate deeper, we conduct a careful analysis on $\Delta$, and we are surprised to

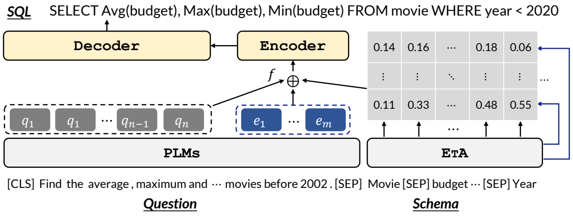

SQL
SELECT Avg(budget), Max(budget), Min(budget) FROM movie WHERE year < 2020
Decoder
Encoder
f
⊕
0.14 0.16 ··· 0.18 0.06
0.11 0.33 ··· 0.48 0.55
$q_1$ $q_1$ ··· $q_{n-1}$ $q_n$
$e_1$ ··· $e_m$
PLMs
ETA
[CLS] Find the average , maximum and ··· movies before 2002 . [SEP] Movie [SEP] budget ··· [SEP] Year
Question
Schema

Figure 4: The illustration of the solution to couple ETA with downstream text-to-SQL parsers.

find that values of $\Delta$ are generally small and not as significantly different with each other as we would expect. Therefore, we believe the success of our approach stems from the fact that it encourages the grounding module to capture subtle differences and strength them.

**RQ3** We apply our approach on BERT-large (BERT$_\text{L}$) and conduct experiments on SPIDER-L. The results show BERT$_\text{L}$ brings an improvement of 2.5% Col$_F$ and 0.5% Tab$_F$, suggesting the possibility of awakening better latent grounding from larger PLMs. Nevertheless, the improvement may also come from more parameters, so the conclusion needs further investigation.

**RQ4** We manually examine 20% of our model's errors on the SQUALL dataset and summarize four main error types: (1) missed grounding - where our model did not ground any token to a concept, (2) technically correct - where our model was technically correct but the annotation was missing, (3) partially correct - where our model did not find all tokens of a concept, (4) wrong grounding - where the model produced incorrect grounding. As shown in Table 3, only a small fraction of errors are wrong grounding, indicating that the main challenge of our approach is recall rather than precision.

## 4 Case Study: Text-to-SQL

The ETA model is proposed for general-purpose uses and intends to enhance different downstream semantic parsing models. To verify it, we take the text-to-SQL task as a case study. In this section, we first present a general solution to couple ETA with different text-to-SQL parsers. Then, we conduct experiments on two off-the-shelf parsers to verify the effectiveness of ETA.

### 4.1 Coupling with Text-to-SQL Parsers

Inspired by Lei et al. (2020), we present a general solution to couple ETA with downstream parsers in Figure 4. As shown, we first obtain a schema-aware representation for each question token, by fusing the token representation and its related schema representation according to the latent grounding $\boldsymbol{\alpha}\in\mathbb{R}^{N\times K}$ (gray matrix in Figure 4). Specifically, given a token representation $\mathbf{q}_n$ and all schema representations $\langle \mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_K \rangle$, the schema-aware representation $\tilde{\mathbf{q}}_n$ for $\mathbf{q}_n$ can be computed as:

$$\tilde{\mathbf{q}}_n = \mathbf{q}_n \oplus \sum_k \boldsymbol{\alpha}_{n,k}\, \mathbf{e}_k. \qquad (6)$$

Then we feed every $\tilde{\mathbf{q}}_n$ into a question encoder to generate hidden states, which are attended by a decoder to decode the SQL query. By contributing to the schema-aware representation, ETA is able to prompt the decoder to predict appropriate schemas during decoding. Notably, the encoder and decoder are not limited to specific modules, and we follow the paper settings in subsequent experiments.

| Model | Dev | | Test |
|---|---|---|---|
| | Ex.Match | Ex.Acc | Ex.Acc |
| ALIGN$_\text{P}$ | $37.8 \pm 0.6$ | $56.9 \pm 0.7$ | $46.6 \pm 0.5$ |
| ALIGN$_\text{P}$ + BERT | $44.7 \pm 2.1$ | $63.8 \pm 1.1$ | $51.8 \pm 0.4$ |
| ETA + BERT | $\mathbf{47.6} \pm 2.5$ | $\mathbf{66.6} \pm 1.7$ | $\mathbf{53.8} \pm 0.3$ |
| ALIGN$^\heartsuit$ | $42.2 \pm 1.5$ | $61.3 \pm 0.8$ | $49.7 \pm 0.4$ |
| ALIGN + BERT$^\heartsuit$ | $47.2 \pm 1.2$ | $66.5 \pm 1.2$ | $54.1 \pm 0.2$ |

Table 4: Ex.Match and Ex.Acc results on the dev and test set of WTQ. + BERT means using BERT to enhance encoder. $^\heartsuit$ means the model uses extra schema linking supervision. Both are the same for Table 5.

### 4.2 Experimental Setup

**Datasets and Evaluation** We conduct experiments on two text-to-SQL benchmarks: WikiTableQuestions(WTQ) (Pasupat and Liang, 2015)[5] and Spider (Yu et al., 2018b). Following previous work, we employ three kinds of evaluation metrics: *Exact Match* (Ex.Match), *Exact Set Match* (Ex.Set) and *Execution Accuracy* (Ex.Acc). Ex.Match evaluates the predicted SQL correctness by checking if it is equal to the ground-truth, while Ex.Set evaluates the structural correctness by checking the set match of each SQL clause in the predicted query with respect to the ground-truth. Ex.Acc evaluates the functional correctness of the predicted SQL by checking whether it yields the ground-truth answer.

[5] Note that the original WTQ only contains answer annotations, and here we use the version with SQL annotations provided by Shi et al. (2020). Our training data is a subset of the original train set, while the test data keeps the same.

| Model | Dev | Test |
|---|---|---|
| IRNet + BERT (Guo et al., 2019) | 61.9 | 54.7 |
| IRNet v2 + BERT (Guo et al., 2019) | 63.9 | 55.0 |
| BRIDGE + $\text{BERT}_\text{L}$ (Lin et al., 2020) | 70.0 | 65.0 |
| RATSQL + $\text{BERT}_\text{L}$ (Wang et al., 2020a) | 69.7 | **65.6** |
| $\text{SLSQL}_\text{P}$ + BERT | 57.4 | - |
| $\text{SLSQL}_\text{P}$ + $\text{BERT}_\text{L}$ | 61.0 | - |
| ETA + BERT | 64.5 | 59.5 |
| ETA + $\text{BERT}_\text{L}$ | **70.8** | 65.3 |

Table 5: Ex.Set results on the dev and test set of Spider.

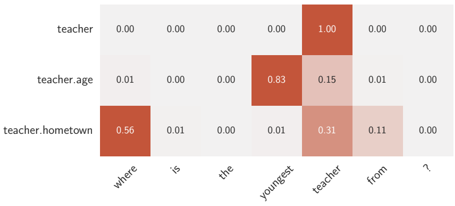


Figure 5: The latent grounding produced by ETA + $\text{BERT}_\text{L}$ for the question "*Where is the youngest teacher from?*".

**Baselines** On WTQ, our baselines include $\text{ALIGN}_\text{P}$ and ALIGN, where the former is a vanilla attention based sequence to sequence model and the latter enhances $\text{ALIGN}_\text{P}$ with an additional schema linking task (Shi et al., 2020). Similarly, on Spider, our main baselines are $\text{SLSQL}_\text{P}$ and its schema linking enhanced version SLSQL (Lei et al., 2020). $\text{SLSQL}_\text{P}$ is made up of a question encoder and a two-step SQL decoder. In the first decoding step, a coarse SQL (i.e., without aggregation functions) is generated. Then the coarse SQL is used to synthesize the final SQL in the second decoding step. Here we also report the performance of SLSQL + BERT (Oracle), where the learnable schema linking module is replaced with human annotations in inference. It represents the maximum potential benefit of schema linking for the text-to-SQL task. Meanwhile, for a comprehensive comparison, we also compare our model with state-of-the-art models on the Spider benchmark[6]. We refer readers to their papers for details.

**Implementation** As for our approach, on WTQ, we employ $\text{ALIGN}_\text{P}$[7] as our base parser, while on Spider we select $\text{SLSQL}_\text{P}$[8] as our base parser. For both parsers, we try to follow the same hyperparameters as described in the paper to reduce other factors that may affect the performance. More implementation details can be found in §C.2.

### 4.3 Experimental Results

Table 4 and Table 5 show the experimental results of several methods on WTQ and Spider respectively. As observed, introducing ETA dramatically improves the performance of both base parsers, demonstrating its effectiveness on downstream tasks. Taking Spider as an illustration, our model ETA + BERT boosts $\text{SLSQL}_\text{P}$ + BERT by an absolute improvement 7.1% on the Ex.Set metric. As the PLM becomes larger (e.g., $\text{BERT}_\text{L}$), the improvement becomes more significant, up to 9.8%. Compared with state-of-the-art methods, our model ETA + $\text{BERT}_\text{L}$ also obtains a competitive performance, which is extremely impressive since it is based on a simple parser.

More interestingly, on both datasets, our model can achieve similar even better performance compared to methods which employ extra grounding supervision. For instance, in comparison with SLSQL + BERT on Spider, our ETA + BERT outperforms it by 3.7%. Taking into account that SLSQL utilizes additional supervision, the performance gain is very surprising. We attribute the gain to two possible reasons: (1) The PLMs already learn latent grounding which is understandable to human experts. (2) Compared with training with strong schema linking supervision, training with weak supervision alleviates the issue of exposure bias, and thus enhance the generalization ability of ETA.

Table 6 presents the model predictions of ETA + $\text{BERT}_\text{L}$ on three real cases. As observed, ETA has learned the grounding about adjective (e.g., oldest → age), entity (e.g., where → hometown) and semantic matching (e.g., registered → student_enrolment). Meanwhile, grounding pairs provide us a useful guide to better understand the model predictions. Figure 5 visualizes the latent grounding for Q2 in Table 6, and more visualization can be found in §D.

## 5 Related Work

The most related work to ours is the line of inducing or probing knowledge in pretrained language models. According to the knowledge category, there are mainly two kinds of methods: one focuses on syntactic knowledge and the other pays attention to semantic knowledge. Under the category of syntac-

[6] https://yale-lily.github.io/spider
[7] https://github.com/tzshi/squall
[8] https://github.com/WING-NUS/slsql

| Question with Alignment | SQL with Alignment |
|---|---|
| 1. Show name$_1$, country$_2$, age$_3$ for all singers$_4$ ordered by age$_3$ from the oldest$_3$ to the youngest. | SELECT name$_1$, country$_2$, age$_3$ FROM singer$_4$ ORDER BY age$_3$ DESC |
| 2. Where$_1$ is the youngest$_2$ teacher$_3$ from? | SELECT hometown$_1$ FROM teacher$_3$ ORDER BY age$_2$ ASC LIMIT 1 |
| 3. For each semester$_1$, what is the name$_2$ and id$_3$ of the one with the most students registered$_4$? | SELECT semester_name$_2$, semester_id$_3$ FROM semesters$_1$ JOIN student_enrolment$_4$ ON semesters.semester_id = student_enrolment.semester_id GROUP BY semester_id$_3$ ORDER BY COUNT(*) DESC LIMIT 1 |

Table 6: The predicted grounding pairs and SQLs of our best model on three real cases from the Spider dev set. The question token and the schema with the same subscript are grounded.

tic knowledge, several work showed that BERT embeddings encoded syntactic information in a structural form and can be recovered (Lin et al., 2019b; Warstadt and Bowman, 2020; Hewitt and Manning, 2019; Wu et al., 2020). However, recent work also showed that BERT did not rely on syntactic information for downstream task performance, and thus doubted the role of syntactic knowledge (Ettinger, 2020; Glavas and Vulic, 2020). As for semantic knowledge, although it is less explored than syntactic knowledge, previous work showed that BERT contained some semantic information, such as entity types (Ettinger, 2020), semantic roles (Tenney et al., 2019) and factual knowledge (Petroni et al., 2019). Different from the above work, we focus on the grounding capability, an under-explored branch of language semantics.

Our work is also closely related to entity linking and schema linking, which can be viewed as subareas of grounding on specific scenarios. Given an utterance, entity linking aims at finding all mentioned entities in it using a knowledge base as candidate pool (Tan et al., 2017; Chen et al., 2018; Li et al., 2020a), while schema linking tries to find all mentioned schemas related to specific databases (Dong et al., 2019; Lei et al., 2020; Shi et al., 2020). Previous work generally either employed full supervision to train linking models (Li et al., 2020a; Lei et al., 2020; Shi et al., 2020), or treated linking as a minor pre-processing(Yu et al., 2018a; Guo et al., 2019; Lin et al., 2019a) and used heuristic rules to obtain the result. Our work is different from them since we optimize the linking model with weak supervision from downstream signals, which is flexible and practicable. Similarly, Dong et al. (2019) utilized downstream supervision to train their linking model. Compared with them using policy gradient, our method is more efficient since it directly learns the grounding module using pseudo alignment as supervision.

## 6 Conclusion & Future Work

In summary, we propose a novel weakly supervised approach to awaken latent grounding from pretrained language models via erasing. Only with downstream signals, our approach can induce latent grounding from pretrained language models which is understandable to human experts. More importantly, we demonstrate that our approach could be applied to off-the-shelf text-to-SQL parsers and significantly improve their performance. For future work, we plan to extend our approach to more downstream tasks such as visual question answering. We also plan to utilize our approach to improve the error locator module in existing interactive semantic parsing systems (Li et al., 2020b).

## Acknowledgement

We would like to thank all the anonymous reviewers for their constructive feedback and useful comments. We also thank Tao Yu and Bo Pang for evaluating our submitted models on the test set of Spider. The first author Qian is supported by the Academic Excellence Foundation of Beihang University for PhD Students.

## Ethical Considerations

This paper conducts experiments on several existing datasets covering the areas of entity linking, schema entity and text-to-SQL. All claims in this paper are based on the experimental results. Every experiment can be conducted on a single Tesla P100 or P40 GPU. No demographic or identity characteristics information is used in this paper.

## References

Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. 2016.

”what is relevant in a text document?”: An interpretable machine learning approach. *CoRR*, abs/1612.07843.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Lihan Chen, Jiaqing Liang, Chenhao Xie, and Yanghua Xiao. 2018. Short text entity linking with fine-grained topics. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM ’18, page 457–466, New York, NY, USA. Association for Computing Machinery.

Sanxing Chen, Aidan San, Xiaodong Liu, and Yangfeng Ji. 2020. A tale of two linkings: Dynamically gating between schema linking and structural linking for text-to-SQL parsing. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2900–2912, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jianpeng Cheng, Siva Reddy, Vijay Saraswat, and Mirella Lapata. 2017. Learning structured natural language representations for semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 44–55, Vancouver, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhen Dong, Shizhao Sun, Hongzhi Liu, Jian-Guang Lou, and Dongmei Zhang. 2019. Data-anonymous encoding for text-to-SQL generation. 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)*, pages 5405–5414, Hong Kong, China. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Goran Glavas and Ivan Vulic. 2020. Is supervised syntactic parsing beneficial for language understanding? an empirical investigation. *CoRR*, abs/2008.06788.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019. Towards complex text-to-SQL in cross-domain database with intermediate representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4524–4535, Florence, Italy. Association for Computational Linguistics.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. 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)*, pages 2733–2743, Hong Kong, China. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on wikisql with table-aware word contextualization. *CoRR*, abs/1902.01069.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. 2020. Re-examining the role of schema linking in text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, Online. Association for Computational Linguistics.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020a. Efficient one-pass end-to-end entity linking for questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6433–6441, Online. Association for Computational Linguistics.

Yuntao Li, Bei Chen, Qian Liu, Yan Gao, Jian-Guang Lou, Yan Zhang, and Dongmei Zhang. 2020b. “what do you mean by that?” a parser-independent interactive approach for enhancing text-to-SQL. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6913–6922, Online. Association for Computational Linguistics.

Percy Liang, Michael I. Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

Kevin Lin, Ben Bogin, Mark Neumann, Jonathan Berant, and Matt Gardner. 2019a. Grammar-based neural text-to-sql generation. *CoRR*, abs/1905.13326.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2020. Bridging textual and tabular data for cross-domain text-to-SQL semantic parsing. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4870–4888, Online. Association for Computational Linguistics.

Yongjie Lin, Yi Chern Tan, and Robert Frank. 2019b. Open sesame: Getting inside BERT’s linguistic knowledge. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 241–253, Florence, Italy. Association for Computational Linguistics.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Qian Liu, Bei Chen, Jiaqi Guo, Jian-Guang Lou, Bin Zhou, and Dongmei Zhang. 2020a. How far are we from effective context modeling? an exploratory study on semantic parsing in context twitter. In *IJCAI*, pages 3580–3586.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020b. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8026–8037. Curran Associates, Inc.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? 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)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Siva Reddy, Oscar Täckström, Michael Collins, Tom Kwiatkowski, Dipanjan Das, Mark Steedman, and Mirella Lapata. 2016. Transforming dependency structures to logical forms for semantic parsing. *Transactions of the Association for Computational Linguistics*, 4:127–140.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

Deb Roy. 2005. Grounding words in perception and action: computational insights. *Trends in Cognitive Sciences*, 9(8):389 – 396.

W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. Müller. 2017. Evaluating the visualization of what a deep neural network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11):2660–2673.

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. 2020. On the potential of lexico-logical alignments for semantic parsing to SQL queries. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1849–1864, Online. Association for Computational Linguistics.

Daniil Sorokin and Iryna Gurevych. 2018. Mixing context granularities for improved entity linking on question answering data across entity categories. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 65–75, New Orleans, Louisiana. Association for Computational Linguistics.

Chuanqi Tan, Furu Wei, Pengjie Ren, Weifeng Lv, and Ming Zhou. 2017. Entity linking for queries by searching Wikipedia sentences. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 68–77, Copenhagen, Denmark. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R. Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020a. RAT-SQL: Relation-aware schema encoding and linking for text-to-SQL parsers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7567–7578, Online. Association for Computational Linguistics.

Chao Wang and Hui Jiang. 2019. Explicit utilization of general knowledge in machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2263–2272, Florence, Italy. Association for Computational Linguistics.

Kai Wang, Weizhou Shen, Yunyi Yang, Xiaojun Quan, and Rui Wang. 2020b. Relational graph attention network for aspect-based sentiment analysis. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3229–3238, Online. Association for Computational Linguistics.

Alex Warstadt and Samuel R. Bowman. 2020. Can neural networks acquire a structural bias from raw linguistic data? *CoRR*, abs/2007.06761.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176, Online. Association for Computational Linguistics.

Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. 2018a. TypeSQL: Knowledge-based type-aware neural text-to-SQL generation. 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 588–594, New Orleans, Louisiana. Association for Computational Linguistics.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018b. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, Brussels, Belgium. Association for Computational Linguistics.

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 and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2*, pages 1050–1055. AAAI Press / The MIT Press.

Luke Zettlemoyer and Michael Collins. 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*, pages 658–666. AUAI Press.

Luowei Zhou, Yannis Kalantidis, Xinlei Chen, Jason J. Corso, and Marcus Rohrbach. 2019. Grounded video description. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6578–6587. Computer Vision Foundation / IEEE.

Yuke Zhu, Oliver Groth, Michael S. Bernstein, and Li Fei-Fei. 2016. Visual7w: Grounded question answering in images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4995–5004. IEEE Computer Society.

# A Evaluation Details

## A.1 Schema Linking

Let $\Omega_{col}$ be a set $\{(c, q)_i | 1 \leq i \leq N\}$ which contains $N$ gold (column-question token) tuples. Let $\overline{\Omega}_{col}$ be a set $\{(\overline{c}, \overline{q})_j | 1 \leq j \leq M\}$ which contains $M$ predicted (column-question token) tuples. We define the precision(Col$_P$), recall(Col$_R$), F1-score(Col$_F$) as:

$$\frac{|\Gamma_{col}|}{|\overline{\Omega}_{col}|}, \frac{|\Gamma_{col}|}{|\Omega_{col}|}, \frac{2\text{Col}_P\text{Col}_R}{\text{Col}_P + \text{Col}_R}$$

where $\Gamma_{col} = \Omega_{col} \bigcap \overline{\Omega}_{col}$. The definitions of Tab$_P$, Tab$_R$, Tab$_F$ are similar. Note that the result reported in Table 8 of Shi et al. (2020) use a different evaluation metrics. Here we re-evaluate their model by the above mentioned metrics for fair comparison.

## A.2 Entity Linking

Let $\Omega = \{(e, [q_s, q_e])_i | 1 \leq i \leq N\}$ be the gold entity-mention set and $\overline{\Omega} = \{(\overline{e}, [\overline{q_s}, \overline{q_e}])_j | 1 \leq j \leq M\}$ be the predicted entity-mention set, where $e$ is the entity, $q_e, q_s$ are the mention boundaries in the question $q$. In the weak matching setting, a prediction is correct only if the ground-truth entity is identified and the predicted mention boundaries overlap with the ground-truth boundaries. Therefore, the True-Positive prediction set is defined as:

$$\Gamma = \{e | (e, [q_s, q_e]) \in \Omega, (e, [\overline{q_s}, \overline{q_e}]) \in \overline{\Omega}, \\ [q_s, q_e] \bigcap [\overline{q_s}, \overline{q_e}] \neq \emptyset\}.$$

The corresponding precision(Ent$_P$), recall(Ent$_R$) and F1(Ent$_F$) are:

$$\frac{|\Gamma|}{|\overline{\Omega}|}, \frac{|\Gamma|}{|\Omega|}, \frac{2\text{Ent}_P\text{Ent}_R}{\text{Ent}_P + \text{Ent}_R}$$

# B Dataset Statistic

All details of datasets used in this paper are shown in Table 7.

# C Implementation Details

For all experiments, we employ the AdamW optimizer and the default learning rate schedule strategy provided by Transformers library (Wolf et al., 2020).

## C.1 Experiments on Grounding

**SQUALL** We use uncased BERT-base as the encoder. The learning rate is $3 \times 10^{-5}$. The training epoch is 50 with a batch size of 16. The dropout rate and the threshold $\tau$ are set to 0.3 and 0.2 respectively. The training process lasts 6 hours on a single 16GB Tesla P100 GPU.

**SPIDER-L** We implement two versions: uncased BERT-base and uncased BERT-large. For both versions, the learning rate is $5 \times 10^{-5}$ and the training epoch is 50. For BERT-base (BERT-large) version, the batch size and gradient accumulation step are set to 12 (6) and 6 (4). The dropout rate and the threshold $\tau$ are set to 0.3 and 0.2 respectively. As for training time, BERT-base (BERT-large) version is trained on a 24GB Tesla P40 and it takes about 16 (48) hours to finish the training process.

**WebQSP$_{\text{EL}}$& GraphQ$_{\text{EL}}$** Due to the large amount of entity candidates, we first use the candidate retrieval method proposed in (Sorokin and Gurevych, 2018) to reduce the number of candidates. After that, we still can not feed all candidates along with the question due to the maximum encoding length of BERT. Therefore, we divide the candidates into multiple chunks and feed each chunk (along with the question) into BERT sequentially.

In implementation, we use uncased BERT-base as the encoder. The learning rate is $1 \times 10^{-5}$ The training epoch is 50 with a batch size of 16. The dropout rate and the threshold $\tau$ are set to 0.3 and 0.3 respectively. The training procedure finishes within 10 hours on a single Tesla M40 GPU.

## C.2 Experiments on Text-to-SQL

For experiments of the text-to-SQL task, we employ the official code released along with Shi et al. (2020) (on WTQ) and Lei et al. (2020) (on Spider). When coupling ETA with these models, we first produce a one-hot grounding matrix derived by grounding pairs and then feed it into them as described in §4.

**WTQ** We use uncased BERT-base as the encoder. The training epoch is 50 with a batch size of 8. The learning rate is $1 \times 10^{-5}$ for the BERT module and $1 \times 10^{-3}$ for other modules. The dropout rate is set to 0.2. The training process finishes within 16 hours on a single 16GB Tesla P100 GPU.

Meanwhile, we follow the previous work (Shi et al., 2020) to employ 5-fold cross-validation, and

| Dataset | Train | | Dev | | Test | |
|---|---|---|---|---|---|---|
| | #Q | #C | #Q | #C | #Q | #C |
| SQUALL | 9,030 | 19,185 | 2,246 | 4,774 | – | – |
| SPIDER-L | 7,000 | 28,848 | 1,034 | 4,360 | – | – |
| WTQ | 9,030 | – | 2,246 | – | 4,344 | – |
| Spider | 7,000 | – | 1,034 | – | 2,147 | – |
| $\text{WebQSP}_{\text{EL}}$ | 2,974 | 3,242 | – | – | 1,603 | 1,806 |
| $\text{GraphQ}_{\text{EL}}$ | 2,089 | 2,253 | – | – | 2,075 | 2,229 |

Table 7: Statistics for all datasets used in our experiments. For SQUALL and WTQ, we only show the size of Split-0, and details of other splits can be found in Table 8. #Q represents the number of questions, #C represents the number of concepts.

| Split | Train | Dev |
|---|---|---|
| 0 | 9,030 | 2,246 |
| 1 | 9,032 | 2,244 |
| 2 | 9,028 | 2,248 |
| 3 | 8,945 | 2,331 |
| 4 | 9,069 | 2,207 |

Table 8: The size of train set and dev set of five splits on SQUALL and WTQ.

| Split | Dev | | Test |
|---|---|---|---|
| | Ex.Match | Ex.Acc | Ex.Acc |
| 0 | 45.10 | 64.43 | 53.57 |
| 1 | 47.39 | 67.01 | 54.17 |
| 2 | 47.24 | 65.93 | 53.61 |
| 3 | 45.99 | 65.72 | 53.41 |
| 4 | 52.38 | 69.73 | 52.41 |

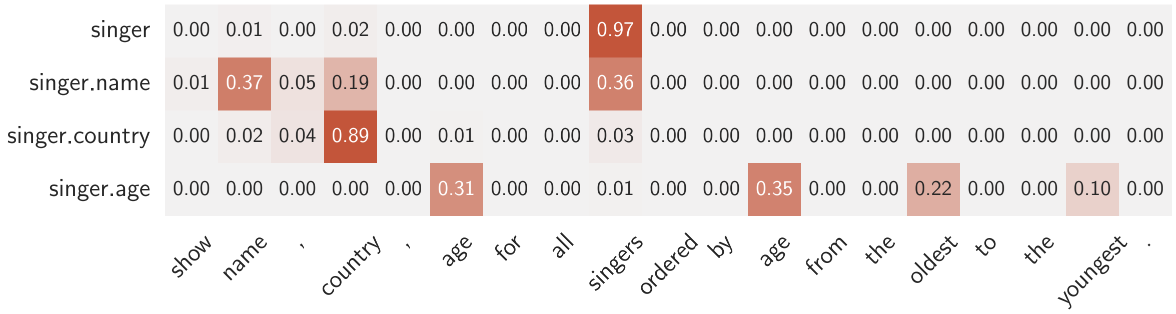
Table 9: The experimental results of all splits on WTQ.

experimental results of all five splits on WTQ using ETA + BERT are shown in Table 9.

**Spider** We implement two versions: uncased BERT-base and uncased BERT-large. For BERT-base (BERT-large), the learning rate is $1.25 \times 10^{-5}$ ($6.25 \times 10^{-6}$) for the BERT module and $1 \times 10^{-4}$ ($5 \times 10^{-5}$) for other modules. The batch size and gradient accumulation step are set to 10 (6) and 5 (4) for BERT-base (BERT-large) version. The dropout rate is set to 0.3. As for training time, BERT-base (BERT-large) version is trained on a 24GB Tesla P40 and it takes about 36 (56) hours to finish the training process.

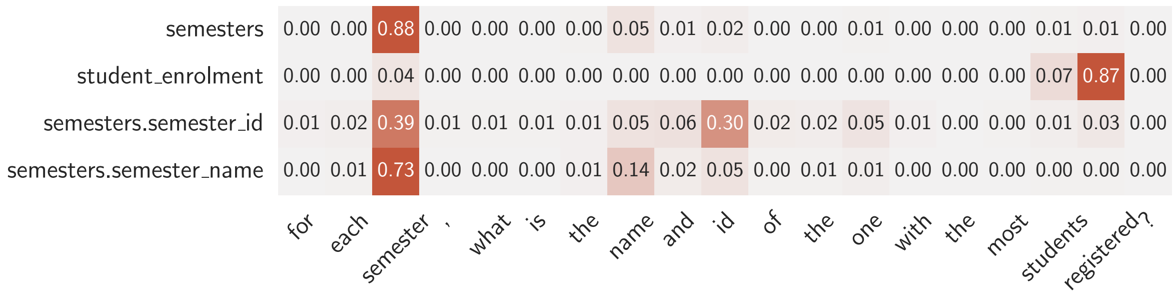
## D Latent Grounding Visualization

Figure 6 and Figure 7 show the latent grounding visualization corresponding to examples in Table 6.



| | show | name | , | country | , | age | for | all | singers | ordered | by | age | from | the | oldest | to | the | youngest | . |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| singer | 0.00 | 0.01 | 0.00 | 0.02 | 0.00 | 0.00 | 0.00 | 0.00 | 0.97 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| singer.name | 0.01 | 0.37 | 0.05 | 0.19 | 0.00 | 0.00 | 0.00 | 0.00 | 0.36 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| singer.country | 0.00 | 0.02 | 0.04 | 0.89 | 0.00 | 0.01 | 0.00 | 0.00 | 0.03 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| singer.age | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.31 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.35 | 0.00 | 0.00 | 0.22 | 0.00 | 0.00 | 0.10 | 0.00 |

Figure 6: The latent grounding produced by ETA + BERT$_L$ for the question "*Show name, country, age for all singers ordered by age from the oldest to the youngest.*".



| | for | each | semester | , | what | is | the | name | and | id | of | the | one | with | the | most | students | registered | ? |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| semesters | 0.00 | 0.00 | 0.88 | 0.00 | 0.00 | 0.00 | 0.00 | 0.05 | 0.01 | 0.02 | 0.00 | 0.00 | 0.01 | 0.00 | 0.00 | 0.00 | 0.01 | 0.01 | 0.00 |
| student_enrolment | 0.00 | 0.00 | 0.04 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.07 | 0.87 | 0.00 |
| semesters.semester_id | 0.01 | 0.02 | 0.39 | 0.01 | 0.01 | 0.01 | 0.01 | 0.05 | 0.06 | 0.30 | 0.02 | 0.02 | 0.05 | 0.01 | 0.00 | 0.00 | 0.01 | 0.03 | 0.00 |
| semesters.semester_name | 0.00 | 0.01 | 0.73 | 0.00 | 0.00 | 0.00 | 0.01 | 0.14 | 0.02 | 0.05 | 0.00 | 0.01 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |

Figure 7: The latent grounding produced by ETA + BERT$_L$ for the question "*For each semester, what is the name and id of the one with the most students registered?*".