Meta-Learning for Domain Generalization in Semantic Parsing

Bailin Wang, Mirella Lapata and **Ivan Titov**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
`bailin.wang@ed.ac.uk, {mlap, ititov}@inf.ed.ac.uk`

Abstract

The importance of building semantic parsers which can be applied to new domains and generate programs unseen at training has long been acknowledged, and datasets testing out-of-domain performance are becoming increasingly available. However, little or no attention has been devoted to learning algorithms or objectives which promote domain generalization, with virtually all existing approaches relying on standard supervised learning. In this work, we use a meta-learning framework which targets zero-shot domain generalization for semantic parsing. We apply a model-agnostic training algorithm that simulates zero-shot parsing by constructing virtual train and test sets from disjoint domains. The learning objective capitalizes on the intuition that gradient steps that improve source-domain performance should also improve target-domain performance, thus encouraging a parser to generalize to unseen target domains. Experimental results on the (English) Spider and Chinese Spider datasets show that the meta-learning objective significantly boosts the performance of a baseline parser.

1 Introduction

Semantic parsing is the task of mapping natural language (NL) utterances to executable programs. While there has been much progress in this area, earlier work has primarily focused on evaluating parsers in-domain (e.g., tables or databases) and often with the same programs as those provided in training (Finegan-Dollak et al., 2018). A much more challenging goal is achieving *domain generalization*, i.e., building parsers which can be successfully applied to new domains and are able to produce complex unseen programs. Achieving this generalization goal would, in principle, let users query arbitrary (semi-)structured data on the Web and reduce the annotation effort required to build multi-domain NL interfaces (e.g., Apple Siri or Amazon Alexa). Current parsers struggle in this setting; for example, we show in Section 5 that a modern parser trained on the challenging Spider dataset (Yu et al., 2018b) has a gap of more than 25% in accuracy between in- and out-of-domain performance. While the importance of domain generalization has been previously acknowledged (Cai and Yates, 2013; Chang et al., 2020), and datasets targetting *zero-shot* (or out-of-domain) performance are becoming increasingly available (Pasupat and Liang, 2015; Wang et al., 2015; Zhong et al., 2017; Yu et al., 2018b), little or no attention has been devoted to studying learning algorithms or objectives which promote domain generalization.

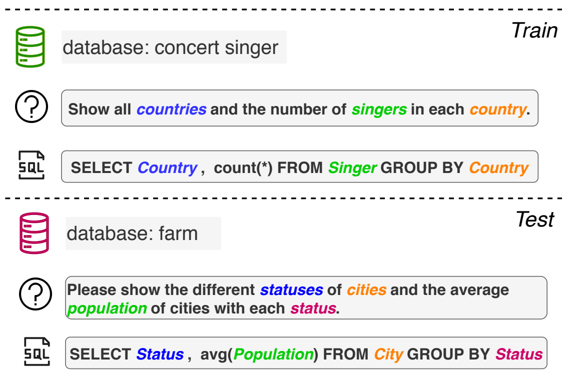

Train
database: concert singer
Show all countries and the number of singers in each country.
SELECT Country , count(*) FROM Singer GROUP BY Country
Test
database: farm
Please show the different statuses of cities and the average population of cities with each status.
SELECT Status , avg(Population) FROM City GROUP BY Status

Figure 1: Zero-shot semantic parsing: at training time, a parser observes instances for the database *concert singer*. At test time, it needs to generate SQL for questions pertaining to the unseen database *farm*.

Conventional supervised learning simply assumes that source- and target-domain data originate from the same distribution, and as a result struggles to capture this notion of domain generalization for zero-shot semantic parsing. Previous approaches (Guo et al., 2019b; Wang et al., 2020; Herzig and Berant, 2018) facilitate domain generalization by incorporating inductive biases in the model, e.g., designing linking features or functions which should be invariant under domain shifts. In this work, we take a different direction and improve the domain generalization of a semantic parser by modifying the learning algorithm and the objective. We draw inspiration from meta-learning (Finn et al., 2017; Li et al., 2018a) and use an objective that optimizes for domain generalization. That is, we consider a set of tasks, where each task is a zero-shot semantic parsing task with its own source and target domains. By optimizing towards better target-domain performance on each task, we encourage a parser to extrapolate from source-domain data and achieve better domain generalization.

Specifically, we focus on text-to-SQL parsing where we aim at translating NL questions to SQL queries and conduct evaluations on unseen databases. Consider the example in Figure 1, a parser needs to process questions to a new database at test time. To simulate this scenario during training, we synthesize a set of virtual zero-shot parsing tasks by sampling disjoint source and target domains[1] for each task from the training domains. The objective we require is that gradient steps computed towards better source-domain performance would also be beneficial to target-domain performance. One can think of the objective as consisting of both the loss on the source domain (as in standard supervised learning) and a regularizer, equal to the dot product between gradients computed on source- and target-domain data. Maximizing this regularizer favours finding model parameters that work not only on the source domain but also generalize to target-domain data. The objective is borrowed from Li et al. (2018a) who adapt a Model-Agnostic Meta-Learning (MAML; Finn et al. 2017) technique for domain generalization in computer vision. In this work, we study the effectiveness of this objective in the context of semantic parsing. This objective is model-agnostic, simple to incorporate and does not require any changes in the parsing model itself. Moreover, it does not introduce new parameters for meta-learning.

Our contributions can be summarized as follows.

- We handle zero-shot semantic parsing by applying a meta-learning objective that *directly* optimizes for domain generalization.

- We propose an approximation of the meta-learning objective that is more efficient and allows more scalable training.

- We perform experiments on two text-to-SQL benchmarks: Spider and Chinese Spider. Our new training objectives obtain significant improvements in accuracy over a baseline parser trained with conventional supervised learning.

- We show that even when parsers are augmented with pre-trained models, e.g., BERT, our method can still effectively improve domain generalization in terms of accuracy.

Our code is available at `https://github.com/berlino/tensor2struct-public`.

[1]We use the terms domain and database interchangeably.

2 Related Work

Zero-Shot Semantic Parsing Developing a parser that can generalize to unseen domains has attracted increased attention in recent years. Previous work has mainly focused on the sub-task of *schema linking* as a means of promoting domain generalization. In schema linking, we need to recognize which columns or tables are mentioned in a question. For example, a parser would decide to select the column *Status* because of the word *statuses* in Figure 1. However, in the setting of zero-shot parsing, columns or tables might be mentioned in a question without ever being observed during training.

One line of work tries to incorporate inductive biases, e.g., domain-invariant n-gram matching features (Guo et al., 2019b; Wang et al., 2020), cross-domain alignment functions (Herzig and Berant, 2018), or auxiliary linking tasks (Chang et al., 2020) to improve schema linking. However, in the cross-lingual setting of Chinese Spider (Min et al., 2019), where questions and schemas are not in the same language, it is not obvious how to design such inductive biases like n-gram matching features. Another line of work relies on large-scale unsupervised pre-training on massive tables (Herzig et al., 2020; Yin et al., 2020) to obtain better representations for both questions and database schemas. Our work is orthogonal to these approaches and can be easily coupled with them. As an example, we show in Section 5 that our training procedure can improve the performance of a parser already enhanced with n-gram matching features (Guo et al., 2019b; Wang et al., 2020).

Our work is similar in spirit to Givoli and Reichart (2019), who also attempts to simulate source and target domains during learning. However, their optimization updates on virtual source and target domains are loosely connected by a two-step training procedure where a parser is first pre-trained on virtual source domains and then fine-tuned on virtual target domains. As we will show in Section 3, our training procedure does not fine-tune on virtual target domains but rather, uses them to evaluate a gradient step (for every batch) on source domains. This is better aligned with what is expected of the parser at test time: there will be no fine-tuning on *real* target domains at test time so there should not be any fine-tuning on *simulated* ones at train time either. Moreover, Givoli and Reichart (2019) treat the division of training domains to virtual train and test domains as a hyper-parameter, which is possible for a handful of domains, but problematic when dealing with hundreds of domains as is the case for text-to-SQL parsing.

Meta-Learning for NLP Meta-learning has been receiving soaring interest in the machine learning community. Unlike conventional supervised learning, meta-learning operates on tasks, instead of data points. Most previous work (Vinyals et al., 2016; Ravi and Larochelle, 2017; Finn et al., 2017) has focused on few-shot learning where meta-learning helps address the problem of learning to learn fast for adaptation to a new task or domain. Applications of meta-learning in NLP are cast in a similar vein and include machine translation (Gu et al., 2018) and relation classification (Obamuyide and Vlachos, 2019). The meta-learning framework however is more general, with the algorithms or underlying ideas applied, e.g., to continual learning (Gupta et al., 2020), semi-supervised learning (Ren et al., 2018), multi-task learning (Yu et al., 2020) and, as in our case, domain generalization (Li et al., 2018a).

Very recently, there have been some applications of MAML to semantic parsing tasks (Huang et al., 2018; Guo et al., 2019a; Sun et al., 2019). These approaches simulate *few-shot* learning scenarios in training by constructing a pseudo-task for each example. Given an example, similar examples are retrieved from the original training set. MAML then encourages strong performance on the retrieved examples after an update on the original example, simulating test-time fine-tuning. Lee et al. (2019) use matching networks (Vinyals et al., 2016) to enable *one-shot* text-to-SQL parsing where tasks for meta-learning are defined by SQL templates, i.e., a parser is expected to generalize to a new SQL template with one example. In contrast, the tasks we construct for meta-learning aim to encourage *generalization* across domains, instead of *adaptation* to a new task with one (or few) examples. One clear difference lies in how meta-train and meta-test sets are constructed. In previous work (e.g., Huang et al. 2018), these come from the *same* domain whereas we simulate domain *shift* and sample different sets of domains for meta-train and meta-test.

Domain Generalization Although the notion of domain generalization has been less explored in semantic parsing, it has been studied in other areas such as computer vision (Ghifary et al., 2015; Zaheer et al., 2017; Li et al., 2018b). Recent work (Li et al., 2018a; Balaji et al., 2018) employed optimization-based meta-learning to handle domain shift issues in domain generalization. We employ the meta-learning objective originally proposed in Li et al. (2018a), where they adapt MAML to encourage generalization in unseen domains (of images). Based on this objective, we propose a cheap alternative that only requires first-order gradients, thus alleviating the overhead of computing second-order derivatives required by MAML.

3 Meta-Learning for Domain Generalization

We first formally define the problem of domain generalization in the context of zero-shot text-to-SQL parsing. Then, we introduce DG-MAML, a training algorithm that helps a parser achieve better domain generalization. Finally, we propose a computationally cheap approximation thereof.

3.1 Problem Definition

Domain Generalization Given a natural language question Q in the context of a relational database $\mathcal{D}$, we aim at generating the corresponding SQL P. In the setting of zero-shot parsing, we have a set of source domains $\mathcal{D}_s$ where labeled question-SQL pairs are available. We aim at developing a parser that can perform well on a set of unseen target domains $\mathcal{D}_t$. We refer to this problem as *domain generalization*.

Parsing Model We assume a parameterized parsing model that specifies a predictive distribution $p_{\boldsymbol{\theta}}(P|Q, \mathcal{D})$ over all possible SQLs. For domain generalization, a parsing model needs to properly condition on its input of questions and databases such that it can generalize well to unseen domains.

Conventional Supervised Learning Assuming that question-SQL pairs from source domains and target domains are sampled i.i.d from the same distribution, the typical training objective of supervised learning is to minimize the loss function of the negative log-likelihood of the gold SQL query:

$$\mathcal{L}_{\mathcal{B}}(\boldsymbol{\theta}) = -\frac{1}{N}\sum_{i=1}^{N} \log p_{\boldsymbol{\theta}}(P|Q, \mathcal{D}) \qquad (1)$$

where N is the size of mini-batch $\mathcal{B}$. Since a mini-batch is randomly sampled from all training source domains $\mathcal{D}_s$, it usually contains question-SQL pairs from a mixture of different domains.

Distribution of Tasks Instead of treating semantic parsing as a conventional supervised learning problem, we take an alternative view based on meta-learning. Basically, wea re interested in a learning algorithm that can benefit from a distribution of choices of source and target domains, denoted by $p(\tau)$, where τ refers to an instance of a zero-shot semantic parsing task that has its own source and target domains.

In practice, we usually have a fixed set of training source domains $\mathcal{D}_s$. We construct a set of virtual tasks τ by randomly sampling disjoint source and target domains from the training domains. Intuitively, we assume that divergences between the test and training domains during the learning phase are representative of differences between training and actual test domains. This is still an assumption, but considerably weaker compared to the i.i.d. assumption used in conventional supervised learning. Next, we introduce the training algorithm called DG-MAML motivated by this assumption.

3.2 Learning to Generalize with DG-MAML

Having simulated source and target domains for each virtual task, we now need a training algorithm that encourages generalization to unseen target domains in each task. For this, we turn to optimization-based meta-learning algorithms (Finn et al., 2017; Nichol et al., 2018; Li et al., 2018a) and apply DG-MAML (Domain Generalization with Model-Agnostic Meta-Learning), a variant of MAML (Finn et al., 2017) for this purpose. Intuitively, DG-MAML encourages the optimization in the source domain to have a positive effect on the target domain as well.

During each learning episode of DG-MAML, we randomly sample a task τ which has its own source domain $\mathcal{D}_s^{\tau}$ and target domain $\mathcal{D}_t^{\tau}$. For the sake of efficiency, we randomly sample mini-batch question-SQL pairs $\mathcal{B}_s$ and $\mathcal{B}_t$ from $\mathcal{D}_s^{\tau}$ and $\mathcal{D}_t^{\tau}$, respectively, for learning in each task. DG-MAML conducts optimization in two steps, namely *meta-train* and *meta-test*.

Meta-Train DG-MAML first optimizes parameters towards better performance in the virtual source domain $\mathcal{D}_s^{\tau}$ by taking one step of stochastic gradient descent (SGD) from the loss under $\mathcal{B}_s$.

$$\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) \qquad (2)$$

where α is a scalar denoting the learning rate of meta-train. This step resembles conventional supervised learning where we use stochastic gradient descent to optimize the parameters.

Meta-Test We then evaluate the resulting parameter $\boldsymbol{\theta}'$ in the virtual target domain $\mathcal{D}_t$ by computing the loss under $\mathcal{B}_t$, which is denoted as $\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}')$.

Our final objective for a task τ is to minimize the joint loss on $\mathcal{D}_s$ and $\mathcal{D}_t$:

$$\begin{aligned}\mathcal{L}_{\tau}(\boldsymbol{\theta}) &= \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}') \\ &= \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}))\end{aligned} \qquad (3)$$

where we optimize towards the better source *and* target domain performance simultaneously. Intuitively, the objective requires that the gradient step conducted in the source domains in Equation (2) be beneficial to the performance of the target domain as well. In comparison, conventional supervised learning, whose objective would be equivalent to $\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta})$, does not pose any constraint on the gradient updates. As we will elaborate shortly, DG-MAML can be viewed as a regularization of gradient updates in addition to the objective of conventional supervised learning.

We summarize our DG-MAML training process in Algorithm 1. Basically, it requires two steps of gradient update (Step 5 and Step 7). Note that $\boldsymbol{\theta}'$ is a function of $\boldsymbol{\theta}$ after the meta-train update. Hence, optimizing $\mathcal{L}_{\tau}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ involves optimizing the gradient update in Equation (2) as well. That is, when we update the parameters $\boldsymbol{\theta}$ in the final update of Step 7, the gradients need to back-propagate though the meta-train updates in Step 5. The update function in Step 7 could be based on any gradient descent algorithm. In this work we use Adam (Kingma and Ba, 2015).

Comment Note that DG-MAML is different from MAML (Finn et al., 2017) which is typically used in the context of few-shot learning. In our case, it encourages domain generalization during training, and does not require an adaptation phase.

Algorithm 1 DG-MAML Training Algorithm

```
Require: Training databases $\mathcal{D}$
Require: Learning rate $\alpha$
1: for step $\leftarrow 1$ to $T$ do
2:     Sample a task $\tau$ of $(\mathcal{D}_s^\tau, \mathcal{D}_t^\tau)$ from $\mathcal{D}$
3:     Sample mini-batch $\mathcal{B}_s^\tau$ from $\mathcal{D}_s^\tau$
4:     Sample mini-batch $\mathcal{B}_t^\tau$ from $\mathcal{D}_t^\tau$
5:     Meta-train update:
         $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s^\tau}(\boldsymbol{\theta})$
6:     Compute meta-test objective:
         $\mathcal{L}_\tau(\boldsymbol{\theta}) = \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}')$
7:     Final Update:
         $\boldsymbol{\theta} \leftarrow \text{Update}(\boldsymbol{\theta}, \nabla_{\boldsymbol{\theta}} \mathcal{L}_\tau(\boldsymbol{\theta}))$
8: end for
```

3.3 Analysis of DG-MAML

To give an intuition of the objective in Equation (3), we follow previous work (Nichol et al., 2018; Li et al., 2018a) and use the first-order Taylor series expansion to approximate it:

$$\begin{aligned}\mathcal{L}_\tau(\boldsymbol{\theta}) =& \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}') \\ =& \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})) \\ \approx& \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}) - \\ & \alpha(\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}))\end{aligned} \tag{4}$$

where in the last step we expand the function $\mathcal{L}_{\mathcal{B}_s}$ at $\boldsymbol{\theta}$. The approximated objective sheds light on what DG-MAML optimizes. In addition to minimizing the losses from both source and target domains, which are $\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta})$, DG-MAML further tries to maximize $\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta})$, the dot product between the gradients of source and target domain. That is, it encourages gradients to generalize between source and target domain within each task τ.

3.4 First-Order Approximation

The final update in Step 7 of Algorithm 1 requires second-order derivatives, which may be problematic, inefficient or non-stable with certain classes of models (Mensch and Blondel, 2018). Hence, we propose an approximation that only requires computing first-order derivatives.

First, the gradient of the objective in Equation (3) can be computed as:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} \mathcal{L}_\tau(\boldsymbol{\theta}) =& \nabla_{\boldsymbol{\theta}} \boldsymbol{\theta}' \nabla_{\boldsymbol{\theta}'} \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}') + \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) \\ =& \big(\boldsymbol{I} - \alpha \nabla^2_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})\big) \nabla_{\boldsymbol{\theta}'} \mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}') \\ & + \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})\end{aligned} \tag{5}$$

where $\boldsymbol{I}$ is an identity matrix and $\nabla^2_{\boldsymbol{\theta}} \mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})$ is the Hessian of $\mathcal{L}_{\mathcal{B}_s}$ at $\boldsymbol{\theta}$. We consider the alternative of ignoring this second-order term and simply assume that $\nabla_{\boldsymbol{\theta}} \boldsymbol{\theta}' = \boldsymbol{I}$. In this variant, we simply combine gradients from source and target domains. We show in the Appendix that this objective can still be viewed as maximizing the dot product of gradients from source and target domain.

The resulting first-order training objective, which we refer to as DG-FMAML, is inspired by Reptile, a first-order meta-learning algorithm (Nichol et al., 2018) for few-shot learning. A two-step Reptile would compute SGD on the same batch twice while DG-FMAML computes SGD on two different batches, $\mathcal{B}_s$ and $\mathcal{B}_t$, once. To put it differently, DG-FMAML tries to encourage *cross-domain* generalization while Reptile encourages *in-domain* generalization.

4 Semantic Parser

In general, DG-MAML is model-agnostic and can be coupled with any semantic parser to improve its domain generalization. In this work, we use a base parser that is based on RAT-SQL (Wang et al., 2020), which currently achieves state-of-the-art performance on Spider.[2]

Formally, RAT-SQL takes as input question Q and schema $\mathcal{S}$ of its corresponding database. Then it produces a program which is represented as an *abstract syntax tree* T in the context-free grammar of SQL (Yin and Neubig, 2018). RAT-SQL adopts the encoder-decoder framework for text-to-SQL parsing. It has three components: an initial encoder, a transformer-based encoder and an LSTM-based decoder. The initial encoder provides initial representations, denoted as Q_{init} and $\mathcal{S}_{init}$ for the question and the schema, respectively. A relation-aware transformer (RAT) module then takes the initial representations and further computes context-aware representations Q_{enc} and $\mathcal{S}_{enc}$ for the question and the schema, respectively. Finally, a decoder generates a sequence of production rules that constitute the abstract syntax tree T based on Q_{enc} and $\mathcal{S}_{enc}$. To obtain Q_{init} and $\mathcal{S}_{init}$, the initial encoder could either be 1) LSTMs (Hochreiter and Schmidhuber, 1997) on top of pre-trained word embeddings, like GloVe (Pennington et al., 2014), or 2) pre-trained contextual embeddings like BERT (Devlin et al.,

[2]We re-implemented RAT-SQL, and added a component for value prediction so that our base parsers can be evaluated by execution accuracy.

2019). In our work, we will test the effectiveness of our method for both variants.

As shown in Wang et al. (2020), the encodings Q_{enc} and $\mathcal{S}_{enc}$, which are the output of the RAT module, heavily rely on schema-linking features. These features are extracted from a heuristic function that links question words to columns and tables based on n-gram matching, and they are readily available in the conventional mono-lingual setting of the Spider dataset. However, we hypothesize that the parser's over-reliance on these features is specific to Spider, where annotators were shown the database schema and asked to formulate queries. As a result, they were prone to re-using terms from the schema verbatim in their questions. This would not be the case in a real-world application where users are unfamiliar with the structure of the underlying database and free to use arbitrary terms which would not necessarily match column or table names (Suhr et al., 2020). Hence, we will also evaluate our parser in the cross-lingual setting where Q and $\mathcal{S}$ are not in the same language, and such features would not be available.

5 Experiments

To evaluate DG-MAML, we integrate it with a base parser and test it on zero-shot text-to-SQL tasks. By designing an in-domain benchmark, we also show that the out-of-domain improvement does not come at the cost of in-domain performance. We also present some analysis to show how DG-MAML affects domain generalization.

5.1 Datasets and Metrics

We evaluate DG-MAML on two zero-shot text-to-SQL benchmarks, namely, (English) Spider (Yu et al., 2018b) and Chinese Spider (Min et al., 2019). Chinese Spider is a Chinese version of Spider that translates all NL questions from English to Chinese and keeps the original English database. It introduces the additional challenge of encoding cross-lingual correspondences between Chinese and English.[3] In both datasets, we report exact set match accuracy, following Yu et al. (2018b). We also report execution accuracy in the Spider dataset.

5.2 Baselines

Two kinds of features are widely used in recent semantic parsers to boost domain generalization: schema-linking features (as mentioned in Section 4) and pre-trained emebddings such as BERT. To show that our method can still achieve additional improvements, we compare with strong baselines that are integrated with schema-linking features and pre-trained embeddings. In the analysis (Section 5.6), we will also show the effect of our method when both features are absent in the base parsers.

5.3 Implementation and Hyperparameters

Our base parser is based on RAT-SQL (Wang et al., 2020), which is implemented in PyTorch (Paszke et al., 2019). For English questions and schemas, we use GloVe (Pennington et al., 2014) and BERT-base (Devlin et al., 2019) as the pre-trained embeddings for encoding. For Chinese questions, we use Tencent embeddings (Song et al., 2018) and Multilingual-BERT (Devlin et al., 2019). In all experiments, we use a batch size of $\mathcal{B}_s = \mathcal{B}_t = 12$ and train for up to 20,000 steps. See the Appendix for details on other hyperparameters.

5.4 Main Results

Our main results on Spider and Chinese Spider are listed in Table 1 and 2, respectively.

Non-BERT Models DG-MAML boosts the performance of non-BERT base parsers on Spider and Chinese Spider by 2.1% and 4.5% respectively, showing its effectiveness in promoting domain generalization. In comparison, the performance margin for DG-MAML is more significant in the cross-lingual setting of Chinese Spider. This is presumably due to the fact that heuristic schema-linking features, which help promote domain generalization for Spider, are not applicable in Chinese Spider. We will present more analysis on this in Section 5.6.

BERT Models Most importantly, improvements on both datasets are not cancelled out when the base parsers are augmented with pre-trained representations. On Spider, the improvements brought by DG-MAML remain roughly the same when the base parser is integrated with BERT-base. As a result, our base parser augmented with BERT-base and DG-MAML achieves the best execution accuracy compared with previous models. On Chinese Spider, DG-MAML helps the base parser with multilingual BERT achieve a substantial improvement. Overall, DG-MAML consistently boosts the performance of the base parser, and is complementary to using pre-trained representations.

[3]Please see the appendix for details of the datasets.

Model	Dev	Test
Set Match Accuracy		
SyntaxSQLNet (Yu et al., 2018a)	18.9	19.7
Global-GNN (Bogin et al., 2019)	52.7	47.4
IRNet (Guo et al., 2019b)	55.4	48.5
RAT-SQL (Wang et al., 2020)	**62.7**	57.2
Our Models		
Base Parser	56.4	-
Base Parser + DG-MAML	58.5	-
With BERT-base:		
SyntaxSQLNet + BERT-base (Guo et al., 2019b)	25.0	25.4
IRNet + BERT-base (Guo et al., 2019b)	61.9	54.7
BRIDGE + BERT-base (Lin et al., 2020)	65.5	58.2
RAT-SQL + BERT-base	66.0$^\diamondsuit$	-
Our Models		
Base Parser + BERT-base	66.8	63.3
Base Parser + BERT-base + DG-MAML	**68.9**	**65.2**
With BERT-large:		
RYANSQL + BERT-large (Choi et al., 2020)	**70.6**	60.6
RAT-SQL + BERT-large (Wang et al., 2020)	69.7	**65.6**
Execution Accuracy		
GAZP + Distil-BERT (Zhong et al., 2020)	59.2	53.5
BRIDGE + BERT-base (Lin et al., 2020)	65.3	59.9
Our Models		
Base Parser + BERT-base	66.8	64.1
Base Parser + BERT-base + DG-MAML	**69.3**	**66.1**

Table 1: Accuracy (%) on the development and test sets of Spider. The first half shows set match accuracy for both non-BERT and BERT models; the second half shows execution accuracy of BERT models. Due to the number of model submissions constraint enforced by the Spider team, we only evaluate our BERT models on the test set for now. The number with $^\diamondsuit$ is produced by running the code of Wang et al. (2020).

5.5 In-Domain vs. Out-of-Domain

To confirm that the base parser struggles when applied out-of-domain, we construct an in-domain setting and measure the gap in performance. This setting also helps us address a natural question: does using DG-MAML hurt in-domain performance? This would not have been surprising as the parser is explicitly optimized towards better performance on unseen target domains.

To answer these questions, we create a new split of Spider. Specifically, for each database from the training and development set of Spider, we include 80% of its question-SQL pairs in the new training set and assign the remaining 20% to the new test set. As a result, the new split consists of 7702 training examples and 1991 test examples. When using this split, the parser is tested on databases that all have been seen during training. We evaluate the non-BERT parsers with the same metric of set match for evaluation.

Does the parser struggle out-of-domain? As in-domain and out-of-domain setting have different splits, and thus do not use the same test set, the direct comparison between them only serves as a proxy to illustrate the effect of domain shift. We show that, despite the original split of out-of-domain setting containing a larger number of training examples (8659 vs 7702), the base parser tested in-domain achieves a much better performance (78.2%) than its counterpart tested out-of-domain (56.4%). This suggests that the domain shift genuinely hurts the base parser.

Model	Dev	Test
SyntaxSQLNet (Yu et al., 2018a)	16.4	13.3
Our Models		
Base Parser	31.0	23.0
Base Parser + DG-MAML	**35.5**	**26.8**
With Multilingual BERT (M-BERT):		
RAT-SQL + M-BERT (Anonymous)	41.4	37.3
RYANSQL + M-BERT (Choi et al., 2020)	41.3	34.7
Our Models		
Base Parser + M-BERT	47.0	44.3
Base Parser + M-BERT + DG-MAML	**50.1**	**46.9**

Table 2: Set match accuracy (%) on the development and test sets of Chinese Spider.

Does DG-MAML hurt in-domain performance? We study DG-MAML in the in-domain setting to see if it hurts in-domain performance. Somewhat surprisingly, we instead observe a modest improvement (+1.1%) over the base parser. This suggests that DG-MAML, despite optimizing the model towards domain generalization, captures, to a certain degree, a more general notion of generalization or robustness, which appears beneficial even in the in-domain setting.

5.6 Additional Experiments and Analysis

We first discuss additional experiments on linking features and DG-FMAML, and then present further analysis probing how DG-MAML works. As the test sets for both datasets are not publicly available, we will use the development sets.

Linking Features As mentioned in Section 2, previous work addressed domain generalization by focusing on the sub-task of schema linking. For Spider, where questions and schemas are both in English, Wang et al. (2020) leverage n-gram matching features which improve schema linking and significantly boost parsing performance. However, in Chinese Spider, it is not easy and obvious how to design such linking heuristics. Moreover, as pointed out by Suhr et al. (2020), the assumption

Model	Dev (%)
Spider	
Base Parser	55.6 ± 0.5
+ DG-FMAML	56.8 ± 1.2
+ DG-MAML	**58.0 ± 0.8**
Base Parser without Features	38.2 ± 1.0
+ DG-FMAML	41.8 ± 1.5
+ DG-MAML	**43.5 ± 0.9**
Chinese Spider	
Base Parser	29.7 ± 1.1
+ DG-FMAML	32.5 ± 1.3
+ DG-MAML	**34.3 ± 0.9**

Table 3: Accuracy (and ±95% confidence interval) on the development sets of Spider and Chinese Spider.

that columns/tables are explicitly mentioned is not general enough, implying that exploiting matching features would not be a good general solution to domain generalization. Hence, we would like to see *whether DG-MAML can be beneficial when those features are not present*.

Specifically, we consider a variant of the base parser that does not use this feature, and train it with conventional supervised learning and with DG-MAML for Spider. As shown[4] in Table 3, we confirm that those features have a big impact on the base parser. More importantly, in the absence of those features, DG-MAML boosts the performance of the base parser by a larger margin. This is consistent with the observation that DG-MAML is more beneficial for Chinese Spider than Spider, in the sense that the parser would need to rely more on DG-MAML when these heuristics are not integrated or not available for domain generalization.

Effect of DG-FMAML We investigate the effect of the first-order approximation in DG-FMAML to see if it would provide a reasonable performance compared with DG-MAML. We evaluate it on the development sets of the two datasets, see Table 3. DG-FMAML consistently boosts the performance of the base parser, although it lags behind DG-MAML. For a fair comparison, we use the same batch size for DG-MAML and DG-FMAML. However, because DG-FMAML uses less memory, it could potentially benefit from a larger batch size. In practice, DG-FMAML is *twice faster* to train than DG-MAML, see Appendix for details.

Model	Precision	Recall	F1
Spider			
Base Parser	70.0	70.4	70.2
Base Parser + DG-MAML	73.8	70.6	**72.1**
Chinese Spider			
Base Parser	61.5	60.4	61.0
Base Parser + DG-MAML	66.8	61.2	**63.9**

Table 4: Performance (%) of column prediction on the development sets of Spider and Chinese Spider.

Probing Domain Generalization Schema linking has been the focus of previous work on zero-shot semantic parsing. We take the opposite direction and use this task to probe the parser to see if it, at least to a certain degree, achieves domain generalization due to improving schema linking. We hypothesize that *improving linking is the mechanism which prevents the parser from being trapped in overfitting the source domains*.

We propose to use 'relevant column recognition' as a probing task. Specifically, relevant columns refer to the columns that are mentioned in SQL queries. For example, the SQL query "*Select Status, avg(Population) From City Groupby Status*" in Figure 1 contains two relevant columns: 'Status' and 'Population'. We formalize this task as a binary classification problem. Given a NL question and a column from the corresponding database, a classifier should predict whether the column is mentioned in the gold SQL query. We hypothesize that representations from the DG-MAML parser will be more predictive of relevance than those of the baseline, and the probing classifier will detect this difference in the quality of the representations.

We first obtain the representations for NL questions and schemas from the parsers and keep them fixed. The binary classifier is then trained based only on these representations. For classifier training we use the same split as the Spider dataset, i.e., the classifier is evaluated on unseen databases. Details of the classifier are provided in the Appendix. The results are shown in Table 4. The classifier trained on the parser with DG-MAML achieves better performance. This confirms our hypothesis that using DG-MAML makes the parser have better encodings of NL questions and database schemas and that this is one of the mechanisms the parsing model uses to ensure generalization.

6 Conclusions

The task of zero-shot semantic parsing has been gaining momentum in recent years. However, previ-

[4]Some results in Table 3 differ from Table 1. The former reports dev set performance over three runs, while the latter shows the best model, selected based on dev set performance.

ous work has not proposed algorithms or objectives that explicitly promote domain generalization. We rely on the meta-learning framework to encourage domain generalization. Instead of learning from individual data points, DG-MAML learns from a set of virtual zero-shot parsing tasks. By optimizing towards better target-domain performance in each simulated task, DG-MAML encourages the parser to generalize better to unseen domains.

We conduct experiments on two zero-shot text-to-SQL parsing datasets. In both cases, using DG-MAML leads to a substantial boost in performance. Furthermore, we show that the faster first-order approximation DG-FMAML can also help a parser achieve better domain generalization.

Acknowledgements

We thank Bo Pang, Tao Yu, Qingkai Min and Yuefeng Shi for their help with the evaluation. We would like to thank the anonymous reviewers for their valuable comments, and Jackie Cheung for pointing out a typo in Eq 4 in the draft version. We gratefully acknowledge the support of the European Research Council (Titov: ERC StG BroadSem 678254; Lapata: ERC CoG TransModal 681760), the Dutch National Science Foundation (NWO VIDI 639.022.518) and EU H2020 Research and Innovation Programme (GoURMET 825299).

References

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. 2018. Metareg: Towards domain generalization using meta-regularization. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 1006–1016.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019. Global reasoning over database structures for text-to-SQL parsing. 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 3659–3664, Hong Kong, China. Association for Computational Linguistics.

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433, Sofia, Bulgaria. Association for Computational Linguistics.

Shuaichen Chang, Pengfei Liu, Yun Tang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Zero-shot text-to-sql learning with auxiliary task. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7488–7495. AAAI Press.

DongHyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin. 2020. Ryansql: Recursively applying sketch-based slot fillings for complex text-to-sql in cross-domain databases. *arXiv preprint arXiv:2004.03125*.

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.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. 2018. Improving text-to-SQL evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 351–360, Melbourne, Australia. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. 2015. Domain generalization for object recognition with multi-task autoencoders. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2551–2559. IEEE Computer Society.

Ofer Givoli and Roi Reichart. 2019. Zero-shot semantic parsing for instructions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4454–4464, Florence, Italy. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. 2019a. Coupling retrieval and meta-learning for context-dependent semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 855–866, Florence, Italy. Association for Computational Linguistics.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. 2019b. 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.

Gunshi Gupta, Karmesh Yadav, and Liam Paull. 2020. La-maml: Look-ahead meta learning for continual learning. *arXiv preprint arXiv:2007.13904*.

Jonathan Herzig and Jonathan Berant. 2018. Decoupling structure and lexicon for zero-shot semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1619–1629, Brussels, Belgium. Association for Computational Linguistics.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Eisenschlos. 2020. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. 2018. Natural language to structured query generation via meta-learning. 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 732–738, New Orleans, Louisiana. Association for Computational Linguistics.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. 2017. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 963–973.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Dongjun Lee, Jaesik Yoon, Jongyun Song, Sanggil Lee, and Sungroh Yoon. 2019. One-shot learning for text-to-sql generation.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2018a. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3490–3497. AAAI Press.

Fei Li and H. V. Jagadish. 2014. Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*, 8(1):73–84.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. 2018b. Domain generalization with adversarial feature learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5400–5409. IEEE Computer Society.

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.

Arthur Mensch and Mathieu Blondel. 2018. Differentiable dynamic programming for structured prediction and attention. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 3459–3468. PMLR.

Qingkai Min, Yuefeng Shi, and Yue Zhang. 2019. A pilot study for Chinese SQL semantic parsing. 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 3652–3658, Hong Kong, China. Association for Computational Linguistics.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Abiola Obamuyide and Andreas Vlachos. 2019. Model-agnostic meta-learning for relation classification with limited supervision. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5873–5879, Florence, Italy. Association for Computational Linguistics.

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 Köpf, 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 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Ana-Maria Popescu, Oren Etzioni, , and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th International Conference on Intelligent User Interfaces*, pages 149–157.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. 2018. Meta-learning for semi-supervised few-shot classification. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. 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 175–180, New Orleans, Louisiana. Association for Computational Linguistics.

Alane Suhr, Ming-Wei Chang, Peter Shaw, and Kenton Lee. 2020. Exploring unexplored generalization challenges for cross-database semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8372–8388, Online. Association for Computational Linguistics.

Yibo Sun, Duyu Tang, Nan Duan, Yeyun Gong, Xiaocheng Feng, Bing Qin, and Daxin Jiang. 2019. Neural semantic parsing in low-resource settings with back-translation and meta-learning.

Lappoon R. Tang and Raymond J. Mooney. 2000. Automated construction of database interfaces: Intergrating statistical and relational learning for semantic parsing. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 133–141.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. 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.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a semantic parser overnight. 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 1332–1342, Beijing, China. Association for Computational Linguistics.

Navid Yaghmazadeh, Yuepeng Wang, Isil Dillig, and Thomas Dillig. 2017. Sqlizer: Query synthesis from natural language. In *International Conference on Object-Oriented Programming, Systems, Languages, and Applications, ACM*, pages 63:1–63:26.

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

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online. Association for Computational Linguistics.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. 2018a. SyntaxSQLNet: Syntax tree networks for complex and cross-domain text-to-SQL task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1653–1663, Brussels, Belgium. 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.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *NeurIPS*.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. 2017. Deep sets. 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 3391–3401.

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

Victor Zhong, Mike Lewis, Sida I. Wang, and Luke Zettlemoyer. 2020. Grounded adaptation for zero-shot executable semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6869–6882, Online. Association for Computational Linguistics.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Analysis of DG-FMAML

Similarly, we use the first-order Taylor expansion to analyze the gradients of DG-FMAML:

$$\begin{aligned}
&\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{T}}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}') \\
=&\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta} - \alpha\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})) \\
\approx&\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}) + \\
&\quad \alpha\nabla^2_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) \\
=&\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}) + \\
&\quad \alpha\nabla_{\boldsymbol{\theta}'}\left(\nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})\right)
\end{aligned}$$

where in Step 3 we expand the gradient function $\nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}$ at $\boldsymbol{\theta}$. In DG-FMAML, there is no gradients back-propogating from $\boldsymbol{\theta}'$ to $\boldsymbol{\theta}$, so we can treat $\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta})$ and $\nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta}')$ as two independent functions with $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ denoting their parameters respectively.

In Step 4, the first two terms $\nabla_{\boldsymbol{\theta}}\mathcal{L}_{\mathcal{B}_s}(\boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}'}\mathcal{L}_{\mathcal{B}_t}(\boldsymbol{\theta})$ can be viewed as the gradient of applying $\boldsymbol{\theta}$ to both source and target domains. The last term can be viewed as maximize the dot product between gradients of source and target domain with respect to $\boldsymbol{\theta}'$. In the same spirit as DG-MAML, DG-FMAML also tries to encourage the gradients to generalize between source and target domains.

B Datasets

Spider Spider consists of 10,181 examples (questions and SQL pairs) from 206 databases, including 1,659 examples taken from the Restaurants (Popescu et al., 2003; Tang and Mooney, 2000), GeoQuery (Zelle and Mooney, 1996), Scholar (Iyer et al., 2017), Academic (Li and Jagadish, 2014), Yelp and IMDB (Yaghmazadeh et al., 2017) datasets. We follow their split and use 8,659 examples (from 146 databases) for training, and 1,034 examples (from 20 databases) as our development set. The remaining 2,147 examples from 40 test databases are held out and kept by the authors for evaluation.

Chinese Spider Chinese Spider is a Chinese version of Spider that translates all NL questions from English to Chinese and keeps the original English database. It simulates the real-life scenario where schemas for most relational databases in industry are written in English while NL questions from users could be in any other language. Following Min et al. (2019), we use the same split of train/development/test as the Spider dataset.

C Hyperparameters

Base Parser We stack 6 relation-aware self-attention layers for encoding. Within them, we set the number of attention heads to be 8 and use dropout rate 0.1. Word embeddings for English questions, column and table names are shared and held fixed except for the 50 most common words in the training set. Word embeddings for Chinese questions are also fixed, except for the 50 most common words in the training set. As noted in Wang et al. (2020), RAT-SQL went through an extensive hyperparameter sweep for non-BERT RAT-SQL, which partially explains why our non-BERT base parser is not as good as it in Spider. However, after the integration of BERT representations, our base parser slightly outperforms RAT-SQL, as shown in the main paper.

Preprocessing A major difference between our base parser and RAT-SQL (Wang et al., 2020) is the way of preprocessing. During preprocessing, input questions, column names and table names in schemas are tokenized and lemmatized by Stanza (Qi et al., 2020) which can handle both English and Chinese.

Learning Rates We use the learning rate of $\alpha = 5 \times 10^{-4}$ for meta-train. For the final update of parameters, we use Adam (Kingma and Ba, 2015) with the learning rate 6×10^{-4}. We manually search for the best meta-train learning rates from 1×10^{-4} to 9×10^{-4} with the step size 2×10^{-4}, based on performance on the development set. Other hyperparameters are not tuned. For the learning rate of final update (not α of meta-train), we use the same scheduler as Wang et al. (2020). Specifically, during the first 500 steps, the learning rate linearly increases from 0 to 6×10^{-4}. Then, it is annealed to 0 with $6 \times 10^{-4}(1 - \frac{step-500}{9500})^{-0.5}$.

Hardware and Model Size Our non-BERT models are trained using NVIDIA GeForce RTX 2080, which has a memory size of 11GB. The base parser has around 10 million parameters, where around 1.5 million parameters are pre-trained embeddings that are mostly fixed during training. For BERT models, we first find the best hyperparameters using GeForce RTX 2080 with a small batch size; then we train them using V100 to save cost.

Average Runtime The average training time for the non-BERT base parser, DG-MAML and DG-FMAML are 10, 24, 13 hours per run. For BERT models, the numbers are 36, 68, 42 hours per run.

C.1 Loss Curve

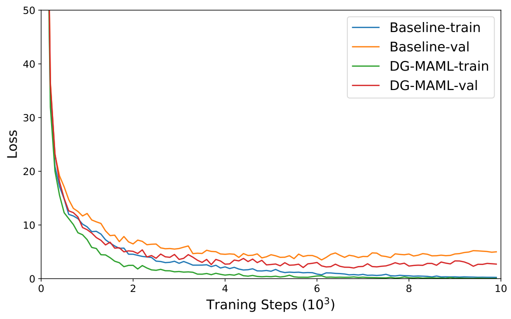
In Figure 2, we show the loss curves of the models on the two datasets during training. In comparison, DG-MAML helps to reduce the gap between training and validation loss.

D Classifier for Probing Domain Generalization

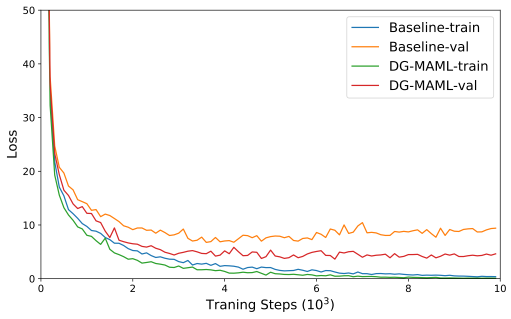
The classifier takes the input of a pair of (column, question) and outputs a binary label indicating whether the column is relevant. As explained in the paper, we retrieve the representations of columns and questions from a pre-trained parser. We denote the representation of a column as $\boldsymbol{c} \in \mathbb{R}^k$, and the representation of a question as $\boldsymbol{q} \in \mathbb{R}^{n \times k}$ where n is the number of words in the question and k is the size of encoding.

For each pair of $(\boldsymbol{c}, \boldsymbol{q})$, we first align the column $\boldsymbol{c}$ softly with the question $\boldsymbol{q}$ using an attention function, and obtain an aligned representation $\boldsymbol{t}$ for the column. Then we compute a score of relevance based on the aligned representation. Finally, a probability p of relevance is computed through a sigmoid function σ.

$$\begin{aligned} \boldsymbol{t} &= \text{Attention}(\boldsymbol{c}, \boldsymbol{q}) \\ score &= \text{MLP}(\boldsymbol{c}, \boldsymbol{t}) \\ p &= \sigma(score) \end{aligned} \tag{6}$$



(a) Loss curves on the Spider dataset.



(b) Loss curves on the Chinese Spider dataset.

Figure 2: Comparison of losses when the parser is trainined with conventaional supervised learning (Baseline) and DG-MAML.