One Semantic Parser to Parse Them All: Sequence to Sequence Multi-Task Learning on Semantic Parsing Datasets

Marco Damonte **Emilio Monti**
Amazon Alexa AI
{dammarco,monti}@amazon.com

Abstract

Semantic parsers map natural language utterances to meaning representations. The lack of a single standard for meaning representations led to the creation of a plethora of semantic parsing datasets. To unify different datasets and train a single model for them, we investigate the use of Multi-Task Learning (MTL) architectures. We experiment with five datasets (GEOQUERY, NLMAPS, TOP, OVERNIGHT, AMR). We find that an MTL architecture that shares the entire network across datasets yields competitive or better parsing accuracies than the single-task baselines, while reducing the total number of parameters by 68%. We further provide evidence that MTL has also better compositional generalization than single-task models. We also present a comparison of task sampling methods and propose a competitive alternative to widespread proportional sampling strategies.

1 Introduction

Semantic parsing is the task of converting natural language into a meaning representation language (MRL). The commercial success of personal assistants, that are required to understand language, has contributed to a growing interest in semantic parsing. A typical use case for personal assistants is Question Answering (Q&A): the output of a semantic parser is a data structure that represents the underlying meaning of a given question. This data structure can be compiled into a query to retrieve the correct answer. The lack of a single standard for meaning representations resulted in the creation of a plethora of semantic parsing datasets, which differ in size, domain, style, complexity, and in the formalism used as an MRL. These datasets are expensive to create, as they normally require expert annotators. Consequently, the datasets are often limited in size.

Multi-task Learning (MTL; Caruana 1997) refers to jointly learning several tasks while sharing parameters between them. In this paper, we use MTL to demonstrate that it is possible to unify these smaller datasets together to train a single model that can be used to parse sentences in any of the MRLs that appear in the data. We experiment with several Q&A semantic parsing dataset for English: GEOQUERY (Zelle and Mooney, 1996), NLMAPS V2 (Lawrence and Riezler, 2018b), TOP (Gupta et al., 2018), and OVERNIGHT (Wang et al., 2015b). In order to investigate the impact of less related tasks, we also experiment on a non-Q&A semantic parsing dataset, targeting a broader coverage meaning representation: AMR (Banarescu et al., 2013), which contains sentences from sources such as broadcasts, newswire, and discussion forums.

Our baseline parsing architecture is a reimplementation of the sequence to sequence model by Rongali et al. (2020), which can be applied to any parsing task as long as the MRL can be expressed as a sequence. Inspired by Fan et al. (2017), we experimented with two MTL architectures: 1-TO-N, where we share the encoder but not the decoder, and 1-TO-1, where we share the entire network. Previous work (Ruder, 2017; Collobert and Weston, 2008; Hershcovich et al., 2018) has focussed on a lesser degree of sharing more closely resembling the 1-TO-N architecture, but we found 1-TO-1 to consistently work better in our experiments.

In this paper we demonstrate that the 1-TO-1 architecture can be used to achieve competitive parsing accuracies for our heterogeneous set of semantic parsing datasets, while reducing the total number of parameters by 68%, overfitting less, and improving on a compositional generalization benchmark (Keysers et al., 2019).

We further perform an extensive analysis of alternative strategies to sample tasks during training. A

Proceedings of the 10th Conference on Lexical and Computational Semantics, pages 173–184
August 5–6, 2021, Bangkok, Thailand (online) ©2021 Association for Computational Linguistics

number of methods to sample tasks proportionally to data sizes have been recently proposed (Wang et al., 2019b; Sanh et al., 2019; Wang et al., 2019a; Stickland and Murray, 2019), which are often used as de facto standards for sampling strategies. These methods rely on the hypothesis that sampling proportionally to the task sizes avoids overfitting the smaller tasks. We show that this hypothesis is not generally verified by comparing proportional methods with an inversely proportional sampling method, and a method based on the per-task loss during training. Our comparison shows that there is not a method that is consistently superior to the others across architectures and datasets. We argue that the sampling method should be chosen as another hyper-parameter of the model, specific to a problem and a training setup.

We finally run experiments on dataset pairs, resulting in 40 distinct settings, to investigate which datasets are most helpful to others. Surprisingly, we observe that AMR and GEOQUERY can work well as auxiliary tasks. AMR is the only graph-structured, non Q&A dataset, and was therefore not expected to help as much as more related Q&A datasets. GEOQUERY is the smallest dataset we tested, showing that low-resource datasets can help high-resource ones instead of, more intuitively, the other way around.

2 Sequence to Sequence Multi-Task Learning

MTL refers to machine learning models that sample training examples from multiple tasks and share parameters amongst them. During training, a batch is sampled from one of the tasks and the parameter update only impacts the part of the network relevant to that task.

The architecture for sequence to sequence semantic parsing that we use in this paper consists of an encoder, which converts the input sentence into a latent representation, and a decoder, which converts the latent representation into the output MRL (Jia and Liang, 2016; Konstas et al., 2017; Rongali et al., 2020). While the input to each task is always natural language utterances, each task is in general characterized by a different meaning representation formalism. It, therefore, follows that the input (natural language) varies considerably less than the output (the meaning representation). Parameter sharing can therefore more intuitively happen in the encoder, where we learn parameters that encode a representation of the natural language. Nevertheless, more sharing can also be allowed, by also sharing parts of the decoder (Fan et al., 2017). In this work, we experiment with two MTL architectures, as shown in Figure 1: 1-TO-N, where we share the encoder but not the decoder, and 1-TO-1, where we share the entire network. As different datasets normally use different MRLs, in the 1-TO-1 architecture we also need a mechanism to inform the network of which MRL to generate. We therefore augment the input with a special token that identifies the task, following Johnson et al. (2017).

3 Experimental Setup

In this section, we describe the datasets used, baseline architectures, and training details.

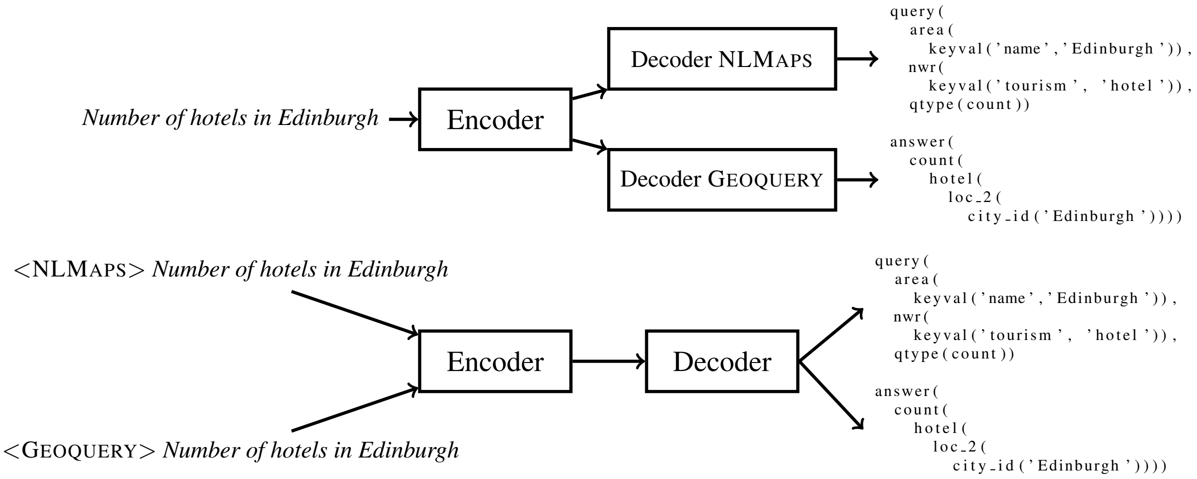
3.1 Data

While we focussed on Q&A semantic parsing datasets, we further consider the AMR dataset in order to investigate the impact of MTL between considerably different datasets. Table 1 shows a training example from each dataset. The sizes of all datasets are shown in Table 2.

Geoquery Questions and queries about US geography (Zelle and Mooney, 1996). The best results on this dataset are reported by Kwiatkowski et al. (2013) via Combinatory Categorial Grammar (Steedman, 1996, 2000) parsing.

NLMaps v2 Questions about geographical facts (Lawrence and Riezler, 2018b), retrieved from OpenStreetMap (Haklay and Weber, 2008). To our knowledge, we are the first to train a parser on the full dataset. Previous work trained a neural parser on a small subset of the dataset and used the rest to experiment with feedback data (Lawrence and Riezler, 2018a). We note that there exists a previous version of the dataset (Haas and Riezler, 2016), for which state-of-the-art results have been achieved with a sequence to sequence approach (Duong et al., 2017). We use the latest version of the dataset due to its larger size.

TOP Navigation and event queries generated by crowdsourced workers (Gupta et al., 2018). The queries are annotated to semantic frames comprising of intents and slots. The best results are achieved by a sequence to sequence model (Aghajanyan et al., 2020).


Number of hotels in Edinburgh
Encoder
Decoder NLMAPS
Decoder GEOQUERY
<NLMAPS> Number of hotels in Edinburgh
<GEOQUERY> Number of hotels in Edinburgh
Encoder
Decoder

```
query(
  area(
    keyval('name','Edinburgh')),
  nwr(
    keyval('tourism', 'hotel')),
  qtype(count))
```

```
answer(
  count(
    hotel(
      loc_2(
        city_id('Edinburgh')))))
```

```
query(
  area(
    keyval('name','Edinburgh')),
  nwr(
    keyval('tourism', 'hotel')),
  qtype(count))
```

```
answer(
  count(
    hotel(
      loc_2(
        city_id('Edinburgh')))))
```

Figure 1: Two MTL architectures for two tasks (A and B): at the top 1-TO-N, where only the encoder is shared; at the bottom 1-TO-1, where we also share the decoder and we add a special token at the beginning of the input sentence.

Dataset	**Input**	**Output**

GEOQUERY	which is the shortest river	`answer(`
		`shortest(`
		`river(all)))`

NLMAPS	name Localities in Nantes	`query(`
		`area(keyval('name','Nantes')),`
		`nwr(keyval('place','locality')),`
		`qtype(findkey('name')))`

TOP	is traffic heavy downtown	`[IN:GET_INFO_TRAFFIC`
		`is traffic heavy in`
		`[SL:LOCATION`
		`downtown]]`

OVERNIGHT	show me all important meetings	`listValue(`
		`filter(`
		`filter(getProperty(`
		`singleton en.meeting)`
		`(string !type))`
		`(string is\_important)))`

AMR	this method will not pollute the environment	`(p / pollute-01 :polarity -`
		`:ARG0 (m / method`
		`:mod (t / this))`
		`:ARG1 (e / environment))`

Table 1: Training examples from each of the datasets used in our experiments. The output logical forms were simplified for the sake of readability.

Dataset	**Train**	**Dev**	**Test**	**Src Vocab**	**Tgt Vocab**
GEOQUERY	540	60	280	279	103
NLMAPS	16172	1843	10594	8628	1012
TOP	28414	4032	8241	11873	116
OVERNIGHT	18781	2093	5224	1921	311
AMR	36521	1368	1371	30169	28880

Table 2: Details of each dataset. "Train", "Dev", and "Test" are the number of examples (questions paired with MRLs) in the training, development, and test splits. "Src Vocab" is the vocabulary size for the input (natural language) and "Tgt Vocab" is the vocabulary size for the output (meaning representation).

Overnight This dataset (Wang et al., 2015b) contains Lambda DCS (Liang, 2013) annotations divided into eight domains: *calendar*, *blocks*, *housing*, *restaurants*, *publications*, *recipes*, *socialnetwork*, and *basketball*. Due to the small size of the domains, we merged them together. The current state-of-the-art results, on single domains, are reported by Su and Yan (2017), who frame the problem as a paraphrasing task. They use denotation (answer) accuracy as a metric, while we report parsing accuracies, a stricter metric.

AMR AMR (Banarescu et al., 2013) has been widely adopted in the semantic parsing community (Artzi et al., 2015; Flanigan et al., 2014; Wang et al., 2015a; Damonte et al., 2017; Titov and Henderson, 2007; Zhang et al., 2019). We used the latest version of the dataset (LDC2017T10), for which the best results were reported by Bevilacqua et al. (2021). The AMR dataset is different from the other datasets, not only in that it is not Q&A, but also in the formalism used to express the meaning representations. While for the other datasets the output logical forms can be represented as trees, in AMR each sentence is annotated as a rooted, directed graph, due to explicit representation of pronominal coreference, coordination, and control structures.

In order to use sequence to sequence architectures on AMR, a preprocessing step is required to remove variables in the annotations and linearize the graphs. In this work, we followed the linearization method by van Noord and Bos (2017).[1]

3.2 Baseline Parser

Our baseline parser is a reimplementation of Rongali et al. (2020): a single-task attentive sequence to sequence model (Bahdanau et al., 2015) with pointer network (Vinyals et al., 2015). The input utterance is embedded with a pretrained RoBERTa encoder (Liu et al., 2019), and subsequently fed into a TRANSFORMER (Vaswani et al., 2017) decoder. The encoder converts the input sequence of tokens $x_1, \ldots, x_n$ into a sequence of context-sensitive embeddings $e_1, \ldots, e_n$. At each time step t, the decoder generates an action a_t. There are two types of actions: output a symbol from the output vocabulary, or output a pointer to one of the input tokens x_i. The final softmax layer provides a probability distribution, for a_t, across all these possible actions. The probability with which we output a pointer to x_i is determined by the attention score on x_i. Finally, we use beam search to find the sequence of actions that maximize the overall output sequence probability.

3.3 Training

All models were trained with Adam (Kingma and Ba, 2014) on P3 AWS machines with one Tesla V100 GPU. To prevent overfitting, we used an early stopping policy to terminate training once the loss on the development set stops decreasing. To account for the effect of the random seed used for initialization, we train three instances of each model with different random seeds. We then report the average and standard deviation on the test set.

We evaluate all Q&A parsing models using the exact match metric, which is computed as the percentage of input sentences that are parsed without any mistake. AMR is instead evaluated using SMATCH (Cai and Knight, 2013), which computes the F1 score of graphs' nodes and edges.[2]

We tuned hyper-parameters for each model based on exact match accuracies on their development sets. While AMR is typically evaluated on SMATCH, to simplify the tuning of our models, we use exact match also for AMR and compute the SMATCH score only for the final models. We performed manual searches (5 trials) for the following hyper-parameters: batch size (10 to 200), learning rate (0.04 to 0.08), number of layers (2 to 6) and units in the decoder (256 to 1024), number of attention heads (1 to 16), and dropout ratio (0.03 to 0.3). For the baseline, we selected the sets of hyper-parameters that maximize performance on the development set of each dataset. To tune the MTL model for each dataset would be costly: we instead selected the set of parameters that maximizes performance on the combination of all development sets. For analogous reasons, when presenting results on MTL between the 40 combinations of dataset pairs, we do not re-tune the models. Final hyper-parameters are shown in Appendix A.

4 Experiments

In Section 4.1, we compare several sampling methods for the 1-TO-1 and 1-TO-N architectures. In Section 4.2 we then compare the MTL models with the single-task baselines. We turn to the issue of

[1] https://github.com/RikVN/AMRwithdefaultsettings

[2] https://github.com/snowblink14/smatch

generalization in Section 4.3, where we use a recently introduced benchmark to evaluate the compositional generalization of our models. Finally, in Section 4.4 we report experiments between dataset pairs to find good auxiliary tasks.

4.1 Task Sampling

As discussed in Section 2, each training batch is sampled from one of the tasks. A simple sampling strategy is to pick the task uniformly, i.e., a training batch is extracted from task t with probability $p_t = 1/N$, where N is the number of tasks. Due to the considerable differences in the sizes of our datasets, we further investigate the impact of previously proposed sampling strategies that take dataset sizes into account:

- PROPORTIONAL (Wang et al., 2019b; Sanh et al., 2019), where p_t is proportional to the size of the training set of task t: D_t. That is: $p_t = D_t/(\sum_t D_t)$;

- LOGPROPORTIONAL (Wang et al., 2019a), where p_t is proportional to $log(D_t)$;

- SQUAREROOT (Stickland and Murray, 2019), where p_t is proportional to $\sqrt{D_t}$;

- POWER (Wang et al., 2019a), where p_t is proportional to $D_t^{0.75}$;

- ANNEALED (Stickland and Murray, 2019), where p_t is proportional to D_t^{α}, with α decreasing at each epoch. When using proportional sampling methods, smaller tasks can be forgotten or interfered with, especially in the final epochs and when the final layers are shared (Stickland and Murray, 2019). The method can therefore be particularly useful for the 1-TO-1 architecture, where the decoder is shared.

We further test two additional sampling strategies:

- INVERSE, where p_t is proportional to $1/D_t$. The idea behind proportional sampling methods is to avoid overfitting smaller tasks and underfitting larger tasks. However, to the best of our knowledge, this intuitive hypothesis has not been explicitly tested. We test the opposite strategy.

- LOSS, where p_t is proportional to $\mathcal{L}_t$, the loss on the development set for task t. This strategy therefore assigns higher sampling probabilities to harder tasks. This strategy is reminiscent of the active learning-inspired sampling method by Sharma et al. (2017).

The results are shown in Table 3 for 1-TO-N and in Table 4 for 1-TO-1. We note that the choice of a sampling method depends on the MTL architecture and the dataset we want to optimize. The choice appears to be more critical for 1-TO-N than for 1-TO-1: for instance, in the case of NLMAPS, the difference between the best sampling method and the worst is 4.3 for 1-TO-N and only 1.3 for 1-TO-1. This suggests that sampling methods are more relevant to train the dedicated layers. 1-TO-1 appears to work well also with PROPORTIONAL, which is expected to suffer for interference when sharing the final layers (Stickland and Murray, 2019). As expected, ANNEALED, which explicitly addresses interference, works particularly well for 1-TO-1.

We presented INVERSE as a way to test the intuition behind proportional strategies. Given the widespread use of proportional methods, we would expect PROPORTIONAL to largely outperform UNIFORM and INVERSE. We instead observe that in most cases it does not outperform INVERSE, and in some cases underperforms it. For 1-TO-1, it does not even match the results of UNIFORM. These results further suggest that there is not a generally superior sampling method, which should instead be picked as an additional hyper-parameter. They also highlight the need to further investigate sampling methods in MTL. The proposed LOSS method is faster and performs particularly well for 1-TO-N. Henceforth, we use LOSS for 1-TO-N and ANNEALED for 1-TO-1, which maximize the average accuracies across datasets.

4.2 One Semantic Parser to Parse Them All

Table 5 compares the MTL results for the chosen sampling methods with the single-task baselines. We also report state-of-the-art parsing accuracies of each dataset for reference. Note that 1-TO-1 has more parameters than 1-TO-N. This is due to the fact that the increased sharing of 1-TO-1 allowed us to train a larger model with 1024 hidden units instead of 512. In order to more directly compare the two MTL architectures, we also train a smaller 1-TO-1 model (1-TO-1-SMALL), which uses the same number of units as 1-TO-N. The re-

Sampling	Geoquery	NLMaps	TOP	Overnight	AMR	Time
UNIFORM	68.8 (±3.8)	81.4 (±2.6)	84.7 (±0.1)	67.0 (±0.8)	61.4 (±1.7)	22h (±2h)
PROP.	70.5 (±1.9)	82.0 (±0.5)	85.0 (±0.0)	68.1 (±0.4)	63.2 (±0.4)	20h (±2h)
LOGPROP.	70.7 (±1.6)	82.8 (±0.7)	85.2 (±0.1)	68.3 (±0.1)	62.9 (±0.5)	18h(±4h)
SQUAREROOT	71.1 (±2.5)	83.4 (±1.1)	84.7 (±0.0)	67.8 (±0.6)	63.6 (±1.0)	21h (±4h)
POWER	73.5 (±1.4)	84.2 (±0.5)	85.1 (±0.3)	68.3 (±0.4)	64.1 (±0.3)	23h (±7h)
ANNEALED	72.1 (±0.0)	82.1 (±0.2)	85.1 (±0.3)	67.8 (±0.2)	63.0 (±0.6)	19h (±2h)
INVERSE	69.9 (±2.4)	84.3 (±0.8)	84.9 (±0.2)	68.4 (±0.7)	64.2 (±0.7)	20h (±2h)
LOSS	73.3 (±1.9)	85.7 (±0.0)	85.2 (±0.1)	68.9 (±0.2)	64.2 (±0.4)	15h (±2h)

Table 3: Comparison of sampling strategies for the 1-TO-N architecture. We report the average over three runs with different random seeds. The standard deviation is in parentheses. All values reported are exact match, except for AMR, where SMATCH is reported. We also report training times (in hours).

Sampling	Geoquery	NLMaps	TOP	Overnight	AMR	Time
UNIFORM	78.5 (±1.4)	87.2 (±0.2)	86.8 (±0.2)	71.1 (±0.2)	66.7 (±0.5)	21h (±4h)
PROP.	77.7 (±1.0)	86.2 (±0.2)	86.5 (±0.2)	70.6 (±0.2)	65.7 (±0.6)	16h (±1h)
LOGPROP.	78.8 (±1.5)	87.2 (±0.1)	86.6 (±0.1)	71.0 (±0.3)	67.3 (±0.5)	23h (±3h)
SQUAREROOT	78.9 (±1.5)	86.8 (±0.1)	86.7 (±0.2)	70.9 (±0.0)	66.4 (±0.3)	17h (±0h)
POWER	78.9 (±0.6)	86.9 (±0.3)	86.6 (±0.1)	71.2 (±0.6)	67.2 (±0.5)	23h (±2h)
ANNEALED	79.8 (±0.7)	87.1 (±0.1)	86.4 (±0.2)	70.8 (±0.4)	67.7 (±0.3)	26h (±1h)
INVERSE	75.0 (±2.3)	87.3 (±0.4)	86.5 (±0.1)	71.2 (±0.5)	66.5 (±0.7)	20h (±3h)
LOSS	76.5 (±1.4)	87.5 (±0.2)	86.5 (±0.1)	71.1 (±0.1)	64.8 (±0.2)	11h (±3h)

Table 4: Comparison of sampling strategies for the 1-TO-1 architecture.

sults indicate that sharing also the decoder provides generally better results, even for the smaller model. Remarkably, compared to the single-task baseline, 1-TO-1 achieves a 68% reduction in the number of learnable parameters. Smaller models can have positive practical impacts as they decrease memory consumption hence reducing costs and carbon footprint (Schwartz et al., 2019). We accomplish this without sacrificing parsing accuracies, which are competitive and in some cases higher than the baselines. This result is particularly promising, as we purposedly included a heterogeneous set of tasks and we use the same set of hyper-parameters for all of them. We can therefore train a single model with accurate parsing for a wide range of datasets, with fewer parameters.

4.3 Generalization

Table 5 also shows that MTL models are slower to converge. This is due to the regularization effect of training multiple tasks (Ruder, 2017): as the loss on the development set keeps improving, the early stopping policy allows the MTL models to be trained for more epochs, resulting in longer training times. This regularization effect allows MTL to have better generalization (Caruana, 1997; Ruder, 2017). In Figure 2 we compare the single-task TOP baseline against the 1-TO-1 model trained on all datasets and evaluated on TOP. We show training and development accuracies as a function of the epochs. We observe that the baseline overfits earlier (early stopping is triggered earlier) and generalizes less (the gap between dev set and training set is larger) compared to the MTL model.

We further evaluate our models on the CFQ dataset (Keysers et al., 2019), designed to test compositional generalization. The idea behind datasets such as CFQ is to include test examples that contain unseen compositions of primitive elements (such as predicates, entities, and question types). To achieve this, a test set is sampled to maximize the compound divergence with the training set, hence containing unseen compositions (MCD). The dataset also contains a second test set, obtained with a random split. A parser that generalizes well is expected to achieve good results on both test sets. Table 6 shows the results of our MTL model when

Model	Geoquery	NLMaps	TOP	Overnight	AMR	Time	Pars
SOTA	89.0	64.4(±0.1)*	87.1	80.6*	84.5		
Baseline	77.6(±2.2)	87.2(±0.7)	85.3(±0.4)	70.2(±0.9)	67.2(±0.3)	7h(±0h)	721M
1-to-N	73.3(±1.9)	85.7(±0.0)	85.2(±0.1)	68.9(±0.2)	64.2(±0.4)	15h(±2h)	203M
1-to-1	79.8(±0.7)	87.1(±0.1)	86.4(±0.2)	70.8(±0.4)	67.7(±0.3)	26h(±1h)	231M
1-to-1-Small	76.7(±1.4)	85.0(±0.8)	85.9(±0.2)	69.7(±0.8)	64.9(±1.3)	20h(±5h)	169M

Table 5: Results of multitasking between all five datasets, compared to the baseline single-task parsers and state-of-the-art results (SOTA) on these datasets. Pars indicates the total number of parameters (in millions). Results marked with * are not directly comparable, as discussed in Section 3.1.

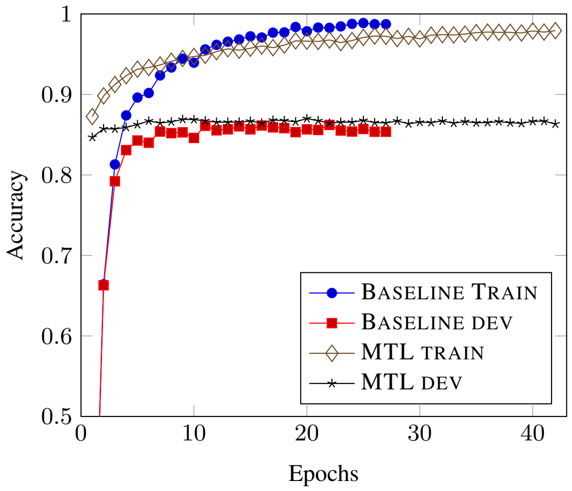


Figure 2: Accuracies on training and dev split at each epoch for the TOP baseline and 1-to-1 MTL parser trained on all datasets and evaluated on TOP.

Model	MCD	Random
Keysers	17.9 (±0.9)	98.5 (±0.2)
Baseline	14.9 (±1.5)	84.9 (±0.7)
1-to-N	16.8 (±0.6)	95.9 (±0.0)
1-to-1	18.9 (±0.8)	95.6 (±0.1)

Table 6: Results on the CFQ dataset. Keysers refers to the results reported by Keysers et al. (2019) for the Transformer model. MCD reports the average of the three released MCD test sets.

adding CFQ as the sixth task.[3] We consider the relative improvements for MCD and Random, as the baseline values are considerably different. We note larger improvements on MCD (+27%) than on Random (+13%) when MTL is used. The results provide initial evidence that the MTL models result in better compositional generalization than the single-task baselines.

4.4 Auxiliary Tasks

Finally, we trained MTL models on dataset pairs to find what datasets are good auxiliary tasks (i.e., tasks that are helpful to other tasks). Note that we do not tune the hyper-parameters of each pairwise model, as we would need to do a costly hyper-parameter search over 40 models. The results are shown in Table 7. The problem of choosing auxiliary tasks has been shown to be challenging (Alonso and Plank, 2016; Bingel and Søgaard, 2017; Hershcovich et al., 2018). Similar to task sampling methods, there is not an easy recipe to choose the auxiliary tasks. However, our results elicit the following surprising observations:

1. AMR is the only dataset to use graph-structured MRL, due to explicit representation of pronominal coreference, coordination, and control structures. It is also the only non-Q&A dataset. Nevertheless, we note that AMR is a competitive auxiliary task, possibly due to its large size and scope. It is also surprising that AMR is often more helpful in the 1-to-1 setup, where the whole network is shared and more related tasks are expected to be preferred.

2. Transfer learning is often used to provide low-resource tasks with additional data from a higher-resource task. However, in our experiments, Geoquery, our smallest dataset, appears to be helpful for the larger TOP dataset.

5 Related Work

A number of alternative meaning representations and semantic parsing datasets have been developed in recent years, spanning from broad-range meaning representations such as Parallel Meaning Bank

[3] For comparison with Keysers et al. (2019), we report mean and 95%-confidence interval radius of 5 runs.

Model		Geoquery	NLMaps	TOP	Overnight	AMR
Baseline		77.6(±2.2)	87.2(±0.7)	85.3(±0.4)	70.2(±0.9)	67.2(±0.3)
1-to-N	+Geoquery	N/A	86.1 (±0.3)	85.8 (±0.0)	69.2 (±0.6)	63.3 (±2.1)
	+NLMaps	77.6 (±0.9)	N/A	85.6 (±0.1)	68.2 (±0.5)	64.4 (±0.5)
	+TOP	79.4 (±0.3)	83.0 (±1.0)	N/A	61.9 (±1.9)	65.3 (±0.5)
	+Overnight	75.7 (±0.8)	85.5 (±0.2)	85.0 (±0.3)	N/A	64.1 (±0.7)
	+AMR	82.0 (±0.4)	85.9 (±0.5)	85.8 (±0.1)	69.0 (±0.4)	N/A
1-to-1	+Geoquery	N/A	87.4 (±0.6)	86.5 (±0.1)	70.9 (±0.8)	66.3 (±0.3)
	+NLMaps	80.0 (±1.8)	N/A	86.4 (±0.3)	69.7 (±1.6)	67.3 (±0.1)
	+TOP	80.5 (±1.5)	85.4 (±0.5)	N/A	65.8 (±1.2)	66.8 (±0.5)
	+Overnight	77.3 (±1.5)	87.0 (±0.3)	86.2 (±0.4)	N/A	67.0 (±0.3)
	+AMR	77.7 (±0.2)	86.9 (±0.3)	86.7 (±0.3)	70.9 (±0.1)	N/A

Table 7: Experiments on dataset pairs. The rows are the auxiliary tasks and the columns are the main tasks.

(Abzianidze et al., 2017) and UCCA (Abend and Rappoport, 2013), to domain-specific datasets such as LCQUAD (Dubey et al., 2019) and KQA Pro (Shi et al., 2020).

Following previous work on semantic parsing (Jia and Liang, 2016; Konstas et al., 2017; Fan et al., 2017; Hershcovich et al., 2018; Rongali et al., 2020), the baseline parser used in this work is based on the popular attentive sequence to sequence framework (Sutskever et al., 2014; Bahdanau et al., 2015). Pointer networks (Vinyals et al., 2015) have demonstrated the importance of decoupling the job of generating new output tokens from that of copying tokens from the input. To achieve this, our models use copy mechanisms, following previous work on semantic parsing (Rongali et al., 2020). We further rely on pre-trained embeddings (Liu et al., 2019).

Compositional generalization has recently attracted attention (Neyshabur et al., 2017; Lake and Baroni, 2018; Finegan-Dollak et al., 2018; Hupkes et al., 2018; Keysers et al., 2019). We used the CFQ dataset (Keysers et al., 2019), with the purpose of assessing their compositional generalization.

MTL (Caruana, 1997; Ruder, 2017) based on sequence to sequence models has been used to address several NLP problems such as syntactic parsing (Luong et al., 2016) and Machine Translation (Dong et al., 2015; Luong et al., 2016). For the task of semantic parsing, MTL has been employed as a way to transfer learning between domains (Damonte et al., 2019) and datasets (Fan et al., 2017; Lindemann et al., 2019; Hershcovich et al., 2018; Lindemann et al., 2019). A shared task on multi-framework semantic parsing with a particular focus on MTL has been recently introduced (Oepen et al., 2019). The 1-to-N and 1-to-1 models have been previously experimented with by Fan et al. (2017), with the latter being an MTL variant of the models used for multilingual parsing by Johnson et al. (2017). An alternative to MTL for transfer learning is based on pre-training on a task and fine-tuning on related tasks (Thrun, 1996). It has been investigated mostly for machine translation tasks (Zoph et al., 2016; Johnson et al., 2017; Bansal et al., 2019) but also for semantic parsing (Damonte et al., 2019).

6 Conclusions

We used MTL to train joint models for a wide range of semantic parsing datasets. We showed that MTL provides large parameter count reduction while maintaining competitive parsing accuracies, even for inherently different datasets. We further discussed how generalization is another advantage of MTL and we used the CFQ dataset to suggest that MTL achieves better compositional generalization. We leave it to future work to further investigate this type of generalization in the context of MTL. We compared several sampling methods, indicating that proportional sampling is not always optimal, showing room for improvements, and introducing a loss-based sampling method as a competitive and promising alternative. We were surprised to see the positive impact of low-resource (Geoquery) and less-related (AMR) datasets can have as auxiliary tasks. Challenges in finding optimal sampling strategies and auxiliary tasks suggest that they should be treated as hyper-parameters to be tuned.

Acknowledgments

The authors would like to thank the three anonymous reviewers for their comments and the Amazon Alexa AI team members for their feedback.

References

Omri Abend and Ari Rappoport. 2013. Universal conceptual cognitive annotation (ucca). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238.

Lasha Abzianidze, Johannes Bjerva, Kilian Evang, Hessel Haagsma, Rik Van Noord, Pierre Ludmann, Duc-Duy Nguyen, and Johan Bos. 2017. The parallel meaning bank: Towards a multilingual corpus of translations annotated with compositional meaning representations. *arXiv preprint arXiv:1702.03964*.

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Mike Haeger, Haoran Li, Yashar Mehdad, Ves Stoyanov, Anuj Kumar, Mike Lewis, et al. 2020. Conversational semantic parsing. *arXiv preprint arXiv:2009.13655*.

Héctor Martínez Alonso and Barbara Plank. 2016. When is multitask learning effective? semantic sequence prediction under varying data conditions. *arXiv preprint arXiv:1612.02251*.

Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. 2015. Broad-coverage CCG semantic parsing with AMR. *Proceedings of EMNLP*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. *Linguistic Annotation Workshop*.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2019. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. 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 58–68.

Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. 2021. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In *Proceedings of AAAI*.

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of EACL*.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. *Proceedings of ACL*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*.

Marco Damonte, Shay B Cohen, and Giorgio Satta. 2017. An incremental parser for abstract meaning representation. In *Proceedings of EACL*.

Marco Damonte, Rahul Goel, and Tagyoung Chung. 2019. Practical semantic parsing for spoken language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 16–23.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. In *Proceedings of ACL*.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. Lc-quad 2.0: A large dataset for complex question answering over wikidata and dbpedia. In *International Semantic Web Conference*, pages 69–78. Springer.

Long Duong, Hadi Afshar, Dominique Estival, Glen Pink, Philip Cohen, and Mark Johnson. 2017. Multilingual semantic parsing and code-switching. In *Proceedings of CoNLL 2017*.

Xing Fan, Emilio Monti, Lambert Mathias, and Markus Dreyer. 2017. Transfer learning for neural semantic parsing. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*.

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.

Jeffrey Flanigan, Sam Thomson, Jaime G Carbonell, Chris Dyer, and Noah A Smith. 2014. A discriminative graph-based parser for the abstract meaning representation. *Proceedings of ACL*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792.

Carolin Haas and Stefan Riezler. 2016. A corpus and semantic parser for multilingual natural language querying of openstreetmap. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 740–750.

Mordechai Haklay and Patrick Weber. 2008. Openstreetmap: User-generated street maps. *Ieee Pervas Comput*, 7(4):12–18.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2018. Multitask parsing across semantic representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 373–385.

Dieuwke Hupkes, Anand Singh, Kris Korrel, German Kruszewski, and Elia Bruni. 2018. Learning compositionally through attentive guidance. *arXiv preprint arXiv:1805.09657*.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. *arXiv preprint arXiv:1606.03622*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. 2019. Measuring compositional generalization: A comprehensive method on realistic data. *arXiv preprint arXiv:1912.09713*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. *Proceedings of ACL*.

Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1545–1556.

Brenden Lake and Marco Baroni. 2018. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2873–2882.

Carolin Lawrence and Stefan Riezler. 2018a. Counterfactual learning from human proofreading feedback for semantic parsing. *arXiv preprint arXiv:1811.12239*.

Carolin Lawrence and Stefan Riezler. 2018b. Improving a neural semantic parser by counterfactual learning from human bandit feedback. *Institute for Computational Linguistics*.

Percy Liang. 2013. Lambda dependency-based compositional semantics. *arXiv preprint arXiv:1309.4408*.

Matthias Lindemann, Jonas Groschwitz, and Alexander Koller. 2019. Compositional semantic parsing across graphbanks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4576–4585.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Thang Luong, Quoc V Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. 2017. Exploring generalization in deep learning. In *Proceedings of NIPS*.

Rik van Noord and Johan Bos. 2017. Dealing with coreference in neural semantic parsing. In *Proceedings of the 2nd Workshop on Semantic Deep Learning*.

Stephan Oepen, Omri Abend, Jan Hajic, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdenka Urešová. 2019. Mrp 2019: Cross-framework meaning representation parsing. In *Proceedings of CoNLL*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. *Proceedings of The Web Conference*.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6949–6956.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2019. Green ai. *arXiv preprint arXiv:1907.10597*.

Sahil Sharma, Ashutosh Jha, Parikshit Hegde, and Balaraman Ravindran. 2017. Learning to multi-task by active sampling. *arXiv preprint arXiv:1702.06053*.

Jiaxin Shi, Shulin Cao, Liangming Pan, Yutong Xiang, Lei Hou, Juanzi Li, Hanwang Zhang, and Bin He. 2020. Kqa pro: A large diagnostic dataset for complex question answering over knowledge base. *arXiv preprint arXiv:2007.03875*.

Mark Steedman. 1996. Surface structure and interpretation. The MIT Press.

Mark Steedman. 2000. The syntactic process. The MIT Press.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995.

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. In *Proceedings of EMNLP*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*.

Sebastian Thrun. 1996. Is learning the n-th thing any easier than learning the first? In *Advances in neural information processing systems*, pages 640–646.

Ivan Titov and James Henderson. 2007. A latent variable model for generative dependency parsing. In *Proceedings of IWPT*.

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

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Proceedings of NIPS*.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, et al. 2019a. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019b. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015a. A transition-based algorithm for AMR parsing. *Proceedings of NAACL*.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015b. Building a semantic parser overnight. In *Proceedings of ACL*.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019. Broad-coverage semantic parsing as transduction. 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 3777–3789.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575.

A Hyper-parameters

Table 8 reports the final hyper-parameters used for our experiments.

Model		**Batch**	**lr**	**Layers**	**Units**	**Heads**	**Dropout**
BASELINE	GEOQUERY	100	0.05	3	512	4	0.1
	NLMAPS	50	0.05	4	512	16	0.05
	TOP	200	0.05	3	512	4	0.04
	OVERNIGHT	10	0.05	3	700	4	0.03
	AMR	10	0.05	4	512	4	0.03
1-TO-N		10	0.05	3	512	4	0.1
1-TO-1		10	0.05	3	1024	4	0.1
1-TO-1-SMALL		10	0.05	3	512	4	0.1

Table 8: Hyper-parameter selected for baselines and MTL models. From left to right the hyper-parameters are: batch size, learning rate, number of layers and units in the decoder, number of attention heads, and dropout ratio.