Improved and Efficient Conversational Slot Labeling through Question Answering

Anonymous ACL submission

Abstract

Transformer-based pretrained language models (PLMs) offer unmatched performance across the majority of natural language understanding (NLU) tasks, including a body of question answering (QA) tasks. We hypothesize that improvements in QA methodology can also be directly exploited in dialog NLU; however, dialog tasks must be *reformatted* into QA tasks. In particular, we focus on modeling and studying *slot labeling* (SL), a crucial component of NLU for dialog, through the QA optics, aiming to improve both its performance and efficiency, and make it more effective and resilient to working with limited task data. To this end, we make a series of contributions: **1)** We demonstrate how QA-tuned PLMs can be applied to the SL task, reaching new state-of-the-art performance, with large gains especially pronounced in such low-data regimes. **2)** We propose to leverage contextual information, required to tackle ambiguous values, simply through natural language. **3)** Efficiency and compactness of QA-oriented fine-tuning are boosted through the use of lightweight yet effective adapter modules. **4)** Trading-off some of the quality of QA datasets for their size, we experiment with larger automatically generated QA datasets for QA-tuning, arriving at even higher performance. Finally, our analysis suggests that our novel QA-based slot labeling models, supported by the PLMs, reach a performance ceiling in high-data regimes, calling for more challenging and more nuanced benchmarks in future work.

1 Introduction and Motivation

Task-oriented conversational systems allow users to interact using natural language to solve well-defined tasks such as restaurant booking, hotel assistance, and travel information (Young, 2002; Raux et al., 2005; Budzianowski et al., 2018). *Slot labeling (SL)*, a crucial component of these systems, aims to fill the correct values associated with predefined *slots* from a domain ontology: e.g., a dialog system for hotel reservations is expected to fill slots such as *check in date* and *the number of guests* with the values extracted from a user utterance (e.g., *next Friday*, *4*). However, the manual construction of such domain ontologies and corresponding annotated examples is expensive, time-consuming, and typically requires domain experts as data designers. For this reason, *few-shot* and *data-efficient* SL has drawn a lot of attention recently (Hou et al., 2020; Henderson and Vulić, 2021; Liu et al., 2020), with the aim to maximize data efficiency by learning from only a handful of task-annotated examples. As for the plethora of other NLP tasks (Qiu et al., 2020; Razumovskaia et al., 2021), these models typically rely on Transformer-based pretrained language models (PLMs) (Devlin et al., 2019; Liu et al., 2019), coupled with SL-specific fine-tuning (Henderson and Vulić, 2021).

In parallel, machine reading comprehension has been fueled by PLM-based improvements and the creation of large-scale datasets (Rajpurkar et al., 2018; Fisch et al., 2019), even matching human-level performance in an array of challenges (Devlin et al., 2019; Zhang et al., 2021). These advances in question answering (QA) models have inspired the ideas of reformatting conversational systems as QA systems (McCann et al., 2018). Such QA-reformatting step can be 'global' (i.e., it can be applied on the full system), or it can be applied to a particular NLU component, as tried quite extensively for dialog state tracking (Gao et al., 2019, 2020; Zhou and Small, 2019).

Recently, Namazifar et al. (2021) have provided preliminary evidence that NLU tasks such as intent detection and slot labeling can also be posed as span-based QA tasks supported by the PLMs: for SL in particular, a question in natural language is defined for each slot, and the answer given by the fine-tuned PLM fills the slot value.[1] Performance

[1] This formulation is very similar to recent work on prompt-

gains of their QA-based NLU methods, especially in low-data scenarios, indicate the suitability of QA methodology for modeling dialog NLU.

Inspired by this emerging line of research, in this paper we propose the **QASL** framework: **Q**uestion **A**nswering for **S**lot **L**abeling, which sheds new light on reformatting SL into QA tasks, and studies it extensively from multiple key aspects, while also aiming to align well with 'real-world' production-ready settings. We summarize these core aspects as follows:

(1) The reformulation of SL into QA allows us to benefit from the adaptation of off-the-shelf PLMs and QA-oriented systems to the dialog domain of interest. Are these adaptations robust across domains and datasets, especially for low-data regimes? Further, are they robust with respect to the chosen PLM and the QA dataset selected for QA-based adaptive fine-tuning (Ruder, 2021)?

(2) To increase efficiency, current span-based SL models only act over the latest user input; however, in some cases, this simplification deteriorates performance as the context of the conversation is necessary to disambiguate between overlapping slots (see Figure 1). How can we adapt QASL to the inherently contextual nature of dialog while maintaining efficiency?

(3) Fully fine-tuning PLMs imposes large training and operational costs, particularly when specialized per-slot SL models are required (Namazifar et al., 2021; Henderson and Vulić, 2021; Mehri and Eskénazi, 2021). Is it possible to build more efficient fine-tuning and adaptation approaches? Can such more lightweight QASL models keep up with the performance of full model fine-tuning?

(4) Can high performance also be obtained with QASL models that leverage larger, automatically generated QA resources for fine-tuning? Can such resources be combined with smaller (but higher-quality) hand-crafted QA resources?

In sum, we push further the understanding of key advantages and limitations of the QA-based approach to dialog SL. The proposed QASL framework is applicable to a wide spectrum of PLMs, and it integrates the contextual information through natural language prompts added to the questions (Figure 1). Experiments conducted on standard SL benchmarks and with different QA-based resources demonstrate the usefulness and robustness

ing task-tuned PLMs (Gao et al., 2021), see also the comprehensive survey on prompting PLMs (Liu et al., 2021).

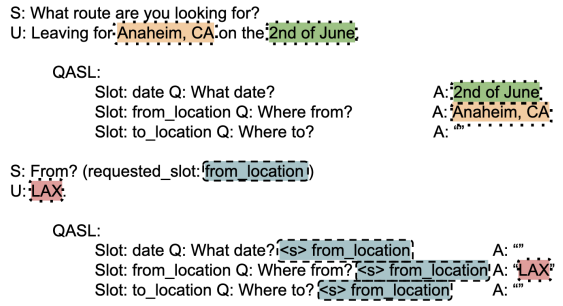

Figure 1: Reformulating slot labeling as QA *with* contextual information. S, U, Q, A denote System, User, Question and Answer (dotted lines), respectively. The requested slot (dashed line), indicated in the previous dialog turns, is added to all questions in the current turn. The second example shows a case where contextual information is crucial for slot disambiguation.

of QASL, with state-of-the-art performance, and most prominent gains observed in low-data scenarios. We also verify the viability of artificially created QA resources for the SL task. Finally, we demonstrate that slot-specific SL models can be fine-tuned with less than 1% parameters of the pre-trained backbone PLM, while maintaining strong SL performance.

2 QASL: Methodology

Preliminaries. Following Namazifar et al. (2021), we pose the SL task as a 'pure' question answering problem. This reformulation into the QA paradigm maps a list of slots S from the domain ontology to a list of corresponding questions Q. For instance, the slots *date*, *from_location*, *to_location*, can be posed as simple natural questions as follows: *"What date?"*, *"Where from?"*, *"Where to?"*, respectively; see Figure 1.[2] At each dialog turn, given the input context C, which may comprise one or more previous turns, the model is sequentially queried with all pre-defined questions appended to C, and returns an answer as a span extracted from the input user utterance, see Figure 1 again.

Fine-Tuning Stages in a Nutshell. We start from any standard Transformer-based (Vaswani et al., 2017) PLM such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), or ELECTRA (Clark et al., 2020). Next, in Stage 1 termed *QA-tuning*, the underlying PLM is fine-tuned with a span-based QA objective using large QA datasets such as SQuAD (Rajpurkar et al., 2018) or MRQA (Fisch et al., 2019). The goal of Stage 1 is to adapt the

[2]The mapping between S and Q can be one-to-many.

model to the span extraction task (Ruder, 2021) with (large and general-purpose) QA data, and this way effectively increase the model's ability to cope with many different questions. Following that, in Stage 2 termed *QASL-tuning*, the model is fine-tuned further for a particular dialog domain. In this stage, the model further specializes to the small subset of in-domain questions that correspond to the slots from the domain ontology.

2.1 QASL with Contextual Information

In complex domains with multiple slots, values can often overlap, which might result in severe prediction ambiguities.[3] The correct prediction can be only made given the context of the conversation.

Moreover, natural conversations are of mixed initiative, where the user can provide more information than it was requested or unexpectedly change the dialog topic (Rastogi et al., 2020). Carrying over the contextual knowledge is a fundamental feature of a successful dialog system (Heck et al., 2020). However, a standard straightforward approach, adopted by the current span-based SL models (Henderson and Vulić, 2021; Namazifar et al., 2021) to boost simplicity and efficiency, runs inference only over the latest user utterance without context or reserves extra parameters for the slots that have been explicitly requested by the system. Put simply, many current approaches discard the potentially crucial contextual information.

In practice, some contextual information from previous dialog turns can be formulated into the so-called *requested slot* (Coope et al., 2020): this means that the current dialog turn is additionally annotated with the slots requested by the system (Coope et al., 2020; Rastogi et al., 2020), helping slot disambiguation. We propose to provide that information to QASL by simply appending the requested slot feature (as a natural language prompt) to the posed question, without any architectural modification, as illustrated in Figure 1. For instance, if the requested slot is present for the slot *arrival_time*, and the current question concerns the slot *date*, the final question takes the following form: *"What dates are you looking for* <s> *arrival time"*, where <s> is a special separator token. When multiple slots are requested, they are all appended to the initial question, each slot separated by one separator token <s>.

2.2 Refining QA-Tuning

Stage 1 of QASL *QA-tuning* is concerned with adaptive transformations of the input PLMs to (general-purpose) span extractors, before the final in-task *QASL-tuning*. We also propose to further refine Stage 1 and divide it into two sub-stages: (a) *Stage 1a* then focuses on fine-tuning on larger but noisier, automatically generated QA datasets, such as PAQ (Lewis et al., 2021); (b) *Stage 1b* continues on the output of Stage 1a, but leverages smaller, manually created and thus higher-quality QA datasets such as SQuAD2.0 (Rajpurkar et al., 2018) and/or MRQA (Fisch et al., 2019).

The rationale behind this refined multi-step *QA-tuning* procedure is that the models 1) should leverage large quantities of automatically generated (QA) data and a task objective aligned with the final task (Henderson and Vulić, 2021), that is, large-scale adaptive fine-tuning (Ruder, 2021) before 2) getting 'polished' (i.e., further specialized towards the final task) leveraging fewer high-quality data. We refer to the QASL model variants which rely on the refined Stage 1 procedure as QASL+.

2.3 Efficient QASL

In principle, one model could be employed to serve all slots in all domains across different deployments. This, however, prevents the separation of different data sources of data, while this is often required from the perspective of data privacy. On the other hand, storing separate slot-specific and domain-specific models derived from heavily parameterized PLMs is extremely storage-inefficient, and their fine-tuning can be prohibitively slow (Henderson and Vulić, 2021).[4] Therefore, with multiple domains and slots, the model compactness and fine-tuning efficiency become crucial features. In order to address these requirements, we rely on and experiment with three different efficiency- and compactness-oriented approaches within the QASL framework in Stage 2, also summarized in Figure 2:

(1) Fine-tuning only the QASL model's *head*, which is responsible for predicting the start and the end of the answer span. All other parameters

[3] For instance, in the domain of restaurant booking, values for the slots *time* and *people* can both be answered with a single number (e.g., *6*) as the only information in the user utterance, causing ambiguity. In another example, Figure 1 shows a conversation from the Buses domain in the DSTC8 dataset (Rastogi et al., 2020); here, it is impossible to distinguish between *from_location* and *to_location* without context.

[4] Distilling PLMs to their smaller counterparts (Lan et al., 2020; Sanh et al., 2019) does not resolve the issue for production-oriented deployments.

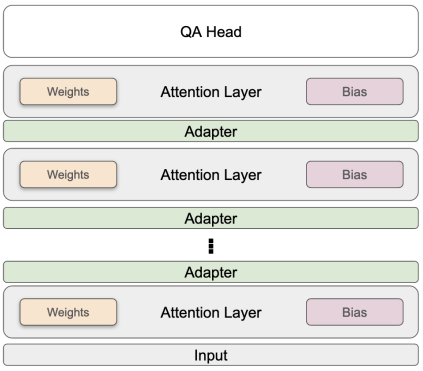

Figure 2: A compact illustration of different efficient fine-tuning schemes used with QASL, and their corresponding parameter subsets (see §2.3).

are kept fixed/frozen. Most QA systems based on PLMs contain a simple one feed-forward layer as the head, using $\leq 0.1\%$ of all the parameters.

(2) Using lightweight tunable bottleneck layers, that is, *adapters* (Houlsby et al., 2019; Pfeiffer et al., 2021), inserted within each Transformer layer of the underlying model. At fine-tuning, only adapter parameters are updated while all the other parameters of the model are kept fixed: i.e., typically $\leq 1\%$ of the PLM's original parameter capacity gets updated (Pfeiffer et al., 2021).

(3) Fine-tuning only *bias* parameters of the attention layers: this approach, termed BitFit (Zaken et al., 2021) in practice fine-tunes less than 0.1% of the full parameters.

It is worth noting that adapters and bias-only tuning (i.e., BitFit) have been evaluated only in full task-data setups in prior work. Here, our use-case scenario adds another layer of complexity as we evaluate them in few-shot scenarios of the SL task.

3 Experimental Setup

Underlying PLMs. We opt for a set of established PLMs with strong performance record on other NLP tasks: RoBERTa (Liu et al., 2019) (its Base and Large variants), and a distilled version of BERT – DistilBERT (Sanh et al., 2019). However, we note that QASL is applicable also to other PLMs.[5]

QA Datasets (Stage 1). We experiment with two manually created QA datasets, (i) SQuAD2.0 (Rajpurkar et al., 2018), and (ii) MRQA (Fisch et al., 2019); and (iii) one automatically generated QA dataset, PAQ (Lewis et al., 2021). SQuAD2.0 was also used in prior work of Namazifar et al. (2021): it consists of 150k QA pairs including 50k negative pairs without any answer. The MRQA dataset is a collection of 18 existing QA datasets, spanning almost 2M QA pairs, converted to the same format of SQuAD2.0. The PAQ dataset, created for open-domain QA, consists of over 65M natural language QA pairs. Due to hardware constraints, we randomly sample two smaller versions from the full PAQ, spanning 5M and 20M QA pairs and denoted as PAQ5 and PAQ20; they are also adapted to the same SQuAD2.0 format.

By selecting these diverse QA-data sources, we validate and compare their usefulness for adaptive QA fine-tuning oriented towards SL, reaching beyond SQuAD2.0 as a standard go-to dataset. We also test if the sheer scale of an automatically generated dataset (i.e., PAQ) can compensate for its lower data quality, compared to manually created SQuAD and MRQA.

Slot Labeling Datasets: Stage 2 and Evaluation. We run experiments on two standard and commonly used SL benchmarks: (i) RESTAURANTS-8k (Coope et al., 2020) and DSTC8 (Rastogi et al., 2020), which are covered by the established DialoGLUE benchmark (Mehri et al., 2020).

RESTAURANTS-8k comprises conversations from a commercial restaurant booking system, and covers 5 slots required for the booking task: *date*, *time*, *people*, *first name*, and *last name*, with a total of 8,198 examples over all 5 slots, see the work of Coope et al. (2020) for further details.

DSTC8 has been introduced during the Dialog System Technology Challenge (DSTC) 8 challenge, and then adapted to the span extraction task by Coope et al. (2020). It includes over 20k annotated multi-domain, task-oriented conversations between humans and a virtual assistant. These conversations involve interactions with services and APIs spanning 4 domains (Buses, Rental Cars, Events, and Homes) and 12 slots; see Rastogi et al. (2020).

Similar to prior work (Coope et al., 2020; Henderson and Vulić, 2021; Mehri and Eskénazi, 2021), we also do tests where we fine-tune on smaller *few-shot data samples* of the two SL datasets, while always evaluating on the same (full) test set. RESTAURANTS-8k comes with 8 different few-shot data samples referred to as $1/128$, $1/64$, $1/32$, $1/16$, $1/8$, $1/4$, $1/2$, 1 (proportions of the full dataset). Similarly, we fine-tune on $1/32$, $1/16$,

[5] For instance, we have run experiments also with ELECTRA (Clark et al., 2020), but do not report its performance as it was consistently outperformed by RoBERTa.

$1/8$, $1/4$, $1/2$, 1 proportions of the full DSTC8.[6]

QASL: Fine-tuning Setup and Hyperparameters. Our QASL implementation is based on the Transformers library (Wolf et al., 2020). Each PLM is equipped with a QA-head which is a feed-forward network with two outputs to compute span start logits and span end logits.

Stage 1 is carried out on 8 V100 GPUs for 2 epochs with 24 QA-pairs per batch per GPU, relying on the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 3e-5. We investigate the following 8 Stage 1 (i.e., QA-tuning) regimes: SQuAD, MRQA, PAQ5, PAQ20 (basic QASL), PAQ5-SQuAD, PAQ5-MRQA, PAQ20-SQuAD, and PAQ20-MRQA (QASL+, see §2.2). The basic Stage 1 setup, unless noted otherwise, is QA-tuning on SQuAD.

Stage 2 (QASL-tuning) proceeds in batches of size 32, again with Adam, and a learning rate 2e-5. All presented results are averaged over 5 different runs. We follow the setup from prior work (Coope et al., 2020; Henderson and Vulić, 2021; Mehri and Eskénazi, 2021), where all the hyper-parameters are fixed across all domains and slots. The reported evaluation metric is the average F1 score across all slots in a given task/domain.[7]

Baselines. We compare QASL against three recent state-of-the-art SL models:[8]

ConVEx (Henderson and Vulić, 2021) defines a novel SL-oriented pretraining objective, termed *pairwise sentence cloze*, combined with SL-tuning of only a subset of parameters. It shows strong performance particularly in few-shot scenarios.

GenSF (Mehri and Eskénazi, 2021) adapts the pre-trained DialoGPT model (Zhang et al., 2020) and steers/constrains its generation freedom to reflect the particular dialog domain; at the same time it adapts the downstream SL task to align better with the architecture of the (fine-tuned) DialoGPT.

QANLU (Namazifar et al., 2021) also reformulates SL as a QA task (see §2) by performing in-task fine-tuning of DistilBERT$_{Base}$ model (Sanh et al., 2019) which was first fine-tuned on SQuAD2.0.

Efficient QASL in Stage 2: Setup. For QA-head-only tuning, the default head in Transformers library (Wolf et al., 2020) is a linear layer of size $[E, 2]$ where E is the size of the output embedding of the PLM. We define the QA head as a feed-forward network with 2 layers, covering $\approx$ 1M parameter. For experiments with adapters, we rely on the lightweight yet effective Pfeiffer architecture (Pfeiffer et al., 2021), using the reduction factor of 16 for all but the first and last Transformers layer, where the factor of 8 was utilized.[9]

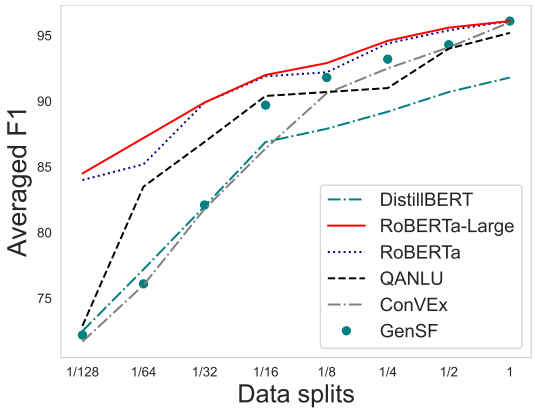

Figure 3: A comparison of slot labeling models on RESTAURANTS-8k. Stage 1 for QASL and QANLU are run on SQuAD2.0. x-axis shows the fraction of the training data used for SL-tuning (see §3).

4 Results and Discussion

QASL versus Baselines. In the first experiment, we benchmark QASL against all baseline models and across different levels of data availability for Stage 2 SL-tuning. We assume SQuAD2.0 as the underlying QA dataset for Stage 1 for all models (including the baseline QANLU), and do not integrate contextual information here (see §2.1). Figure 3 plots the results on RESTAURANTS-8k,[10] and reveals several findings. First, there is an indication that larger models yield performance gains: RoBERTa$_{Large}$ is slightly stronger than RoBERTa$_{Base}$ as the underlying model, although RoBERTa$_{Base}$ also shows very competitive performance across the board. While most models reach very similar and very high performance in the full-data regime, the difference between models becomes much more salient in few-shot setups. The gains in favor of QASL with RoBERTa-s over all baselines are the largest for the scarcest data

[6] The exact numbers are in Appendix A (Table 4).

[7] It is computed with an exact score, that is, the model has to extract exactly the same span as the golden annotation. This is different to a typical QA setup where partial or multiple answers are also taken into account (Rajpurkar et al., 2018).

[8] For full technical details of each baseline model, we refer the reader to their respective papers.

[9] The learning rate has been increased to 1e$-$3 following prior work (Pfeiffer et al., 2021), and it also yielded better performance in our preliminary experiments.

[10] The exact numbers are in Appendix A (Table 5).

scenarios: 1/64 and 1/128.[11] [12]

Using Contextual information. We now investigate if the integration of contextual information in the form of requested slots improves SL performance (see §2.1). Unless noted otherwise, from now on we assume that QASL always integrates the requested slot information. The results on RESTAURANTS-8k for a subset of test examples with non-empty requested slots (i.e., 897 out of all 3,731 test examples), are summarized in Table 1. The variant with requested slot information consistently yields higher F_1 scores, even despite the fact that the test set contains only 86 examples that might cause ambiguity.

The results on the 4 domains of DSTC8, provided in Figure 4 for all test examples, show very similar patterns and improvements over the baseline SL models GenSF and ConVEx, especially in few-shot scenarios. The gains with the contextual variant are less pronounced than in RESTAURANTS-8k as DSTC8 covers a fewer number of ambiguous test examples.

Further, we observe extremely high absolute scores, especially in higher-data setups, which is the first indication that the standard SL benchmarks might become inadequate to distinguish between SL models in the future. We provide a finer-grained analysis of the SL benchmarks later in §5.

Efficient Fine-Tuning in Stage 2. We now proceed with the RoBERTa$_{Base}$ model as our base PLM in all following experiments: it achieves very competitive results while using ≈3 times fewer parameters than RoBERTa$_{Large}$. Table 2 presents the scores obtained with the three efficient fine-tuning approaches (see §2.3) on RESTAURANTS-8k in few-shot scenarios.

Overall, the results indicate that few-shot scenarios are quite challenging for efficient fine-tuning methods, typically evaluated only in full-data scenarios in prior work (Zaken et al., 2021). The adapter-based approach is most effective by far, and is very competitive to full model fine-tuning, even outperforming it in all but the two fewest-data scenarios. The other two efficient approaches fall largely behind in all training setups. In summary, the results empirically validate that adapter-based fine-tuning offers a viable trade-off between performance and efficiency, even in low-data regimes: it fine-tunes only ≈1.5M parameters, translating to 5MB of storage space, compared to 110M parameters (i.e., 550 MB) needed for full fine-tuning.

[11] While we also achieve higher results than QANLU (Namazifar et al., 2021), the exact comparison is not possible since they used different data splits.

[12] We have also tried different question prompts but have not observed any significant variance in the results.

	Without Requested	With Requested
1/128	81.7	**85.8**
1/64	81.0	**87.9**
1/32	86.7	**90.7**
1/16	88.7	**93.8**
1/8	88.9	**95.7**
1/4	91.0	**95.0**
1/2	91.5	**97.0**
1	92.0	**98.0**

Table 1: A comparison of QASL without and with requested slot information on the subset of RESTAURANTS-8k test examples with non-empty requested slots (891 test examples).

	Full	QA head	BitFit	Adapters
1/128	**84.0**	0.0	25.6	81.9
1/64	**85.2**	20.2	27.9	85.0
1/32	89.9	28.3	32.5	**91.0**
1/16	91.9	33.8	52.0	**92.9**
1/8	92.2	40.7	51.7	**93.6**
1/4	94.4	52.6	70.5	**95.2**
1/2	95.4	57.7	88.8	**96.1**
1	96.1	61.8	93.6	**97.0**

Table 2: Average F_1 scores across all slots on the entire RESTAURANTS-8k test data with efficient fine-tuning architectures in Stage 2 (see §2.3), and their comparison to **Full** model fine-tuning.

Different Stage 1 Fine-Tuning Schemes. Note that, until now, the results were based solely on models QA-tuned with SQuAD2.0 in Stage 1. We now test the impact of the QA resource in Stage 1 on the final SL performance. Table 3 presents the results for the 8 Stage 1 regimes (see §3), fine-tuned with QASL on 3 smallest RESTAURANTS-8k training data splits in Stage 2.

When using only one QA dataset in Stage 1, several trends emerge. First, a larger of the two manually created datasets, MRQA, yields consistent gains over SQuAD2.0, over all training data splits. Using larger but automatically created PAQ5 and PAQ20 is on par or even better than using SQuAD, but they cannot match performance with MRQA. This confirms that both QA dataset quality *and* dataset size play an important role in the two-stage adaptation of PLMs into effective slot labellers. Having more PAQ data typically yields worse performance: it seems that more noise from more automatically generated QA pairs gets inserted into the fine-tuning process (cf., PAQ20 versus PAQ5).

However, QASL tuned only with automatically

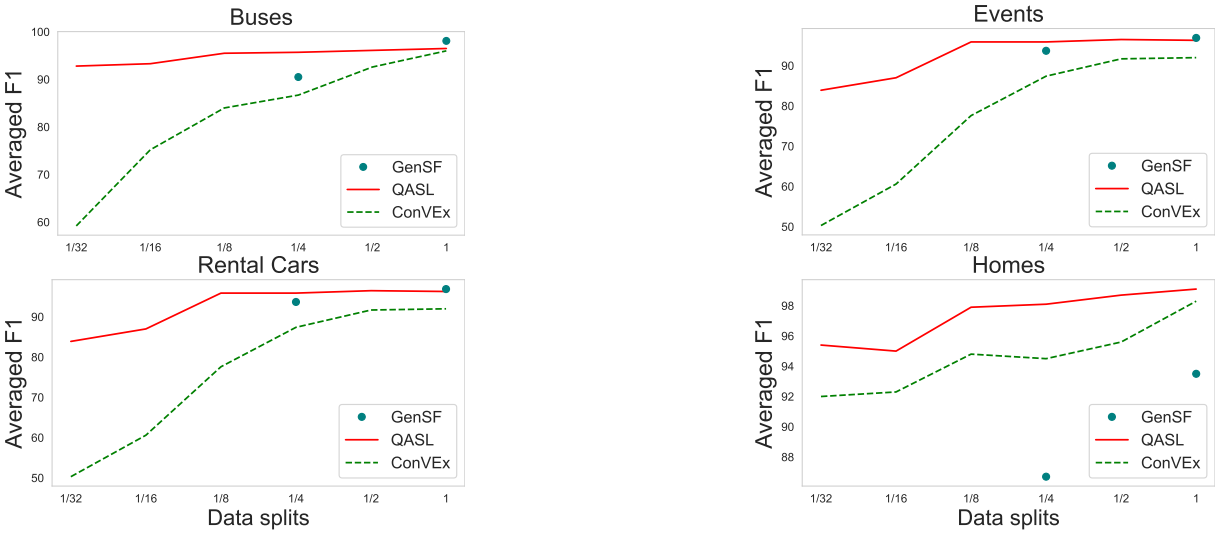

Figure 4: Results on the DSTC8 dataset across 4 domains. The performance of GenSF is taken from the original paper and is only available for two data splits: 1 (full data) and $1/4$. The QASL fine-tunes RoBERTa$_{Large}$ on SQUAD2.0 in Stage 1, and uses contextual requested slot information in Stage 2.

	SQuAD	MRQA	PAQ5	PAQ20	PAQ5-SQuAD	PAQ20-SQuAD	PAQ5-MRQA	PAQ20-MRQA
1/128	84.0	**86.31**	83.62	82.57	86.09	85.19	**86.31**	85.47
1/64	85.2	87.59	86.45	85.64	87.95	87.11	**88.40**	87.65
1/32	89.9	**91.50**	91.14	89.97	91.46	90.92	91.13	91.08

Table 3: F_1 scores over all slots on the RESTAURANTS-8k test data for different QA-tuning regimes in Stage 1.

generated data is still on par or better than tuning with SQuAD2.0. This proves the potential of large-scale (automatically obtained) QA datasets for QA-based slot-labeling in domains that have a small overlap with curated QA data such as SQuAD. The highest gains over SQuAD when using PAQ are obtained for two slots: $first\_name$ and $last\_name$. This stems from the fact that finding the right person's name is a common task with Wikipedia-related corpora. Finally, in two out of the three training data splits, the peak scores are achieved with the refined Stage 1 (the PAQ5-MRQA variant), but the gains of the more expensive PAQ5-MRQA regime over MRQA are mostly inconsequential.

5 SL Data Analysis and Audit

Detected high absolute scores in full-data setups for many models in our comparison (e.g., see Figure 3, Table 2, Figure 4) suggest that the current SL benchmarks might not be able to distinguish between state-of-the-art SL models. The remaining gap to 100% performance might also be due to annotation errors and inconsistencies. We thus inspect the two SL benchmarks in more detail.

On RESTAURANTS-8k, we found that adding the contextual information robustly resolves the issue of ambiguous one-word utterance examples. We identified 86 examples where the utterance is a single number, intentionally meant to test the model's capability of using the requested slot, as they could refer either to *time* or *number of people*. Adding requested slot information eliminates all but 2 of these mistakes. Another challenging group of example concerns rare names - most of the issues come from mixing up *first name* and *last name* since both are requested together.

Upon inspection of RESTAURANTS-8k's test set, we discovered several annotation issues. Analyzed models perform the worst on the *time* slot. This is partly due to the many ways one can express time, but also owning to difficulties in annotations. In the test set, some time examples are in the format *TIME pm*, while others use *TIME p.m.*: in simple words, whether the *pm* postfix is annotated or not is inconsistent. Another inconsistency concerns preposition annotations such as *on*, *at*. In some examples the prepositions are included in the answer (e.g. *is there a table free at 8 in the morning*), in others they are not. A similar challenge concerns annotating 'the' in *date* answers, such as *the first Sunday of September* instead of *first Sunday of September*. This leads the model to select *August 23rd* instead of *the day of August 23rd*. Another an-

notation inconsistency concerns the *people* slot. In some examples, only the concrete number is annotated, other times the noun following is annotated as well: *4 people* vs 4.

A similar analysis of DSTC8 is provided in Appendix B. Given that the cutting-edge SL models are rewarded only if they provide the exact span match (see §3), it seems that they get penalized mostly due to the detected annotation inconsistencies and errors in training and test data. Correcting the inconsistencies would further improve their performance, even to the point of considering the current SL benchmarks 'solved' in their full-data setups. Our simple analysis thus also hints that the community should invest more effort into creating more challenging SL benchmarks in future work.

6 Related Work

Slot Labeling in Dialog. A variety of approaches have been proposed to leverage the semantic knowledge of PLMs like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) for intent classification and dialog state tracking (Chen et al., 2019; Casanueva et al., 2020; Louvan and Magnini, 2020; Gao et al., 2020). The potential of the PLMs has also been exploited in end-to-end multi-domain systems, offering both design simplicity and superior performance over modular systems (Hosseini-Asl et al., 2020; Peng et al., 2021).

The SL task has also benefited from the semantic prowess of PLMs. One family of models employs universal sentence encoders (Devlin et al., 2019) and trains a task-specific head to extract slot value spans (Chao and Lane, 2019; Coope et al., 2020; Rastogi et al., 2020). In more recent work, Henderson and Vulić (2021) define a novel SL-oriented pretraining objective. The proposed model, ConVEx, achieved substantial improvements in the SL task, particularly in low-data regimes. However, contrary to QASL it requires training additional context-related features during fine-tuning. Another line of work relies on reformulating slot labeling as a natural language response generation task by adapting generative language models. Madotto et al. (2020b) shows that this can be done in a zero-shot fashion by priming with task-oriented context. The GenSF model (Mehri and Eskénazi, 2021) adapts the pretrained DialoGPT model for the SL task through constrained generation. These approaches also lack contextualization and do not consider efficiency-oriented fine-tuning.

The work closest to ours is QANLU (Namazifar et al., 2021), which also reformulates SL as a QA task, showing performance gains in low-data regimes. However, QANLU did not incorporate contextual information, did not experiment with different QA resources, nor allowed for efficient and compact fine-tuning.

Efficient Methods in Dialog. Recent dialog work is increasingly interested in the efficiency aspects of both training and fine-tuning. Henderson and Vulić (2021) achieve compactness by fine-tuning only a small subset of decoding layers from the full pretrained model. As mentioned, their ConVEx framework is constrained by the particularities of their pretraining regime and cannot be easily combined with a wealth of different PLMs.

Efficient fine-tuning with easy portability can be achieved by inserting small adapter modules inside pretrained Transformers (Houlsby et al., 2019; Pfeiffer et al., 2021). Adapters make controllable response generation viable for online systems by training task-specific modules per style/topic (Madotto et al., 2020a). Through the adapters injection, Wang et al. (2021); Hung et al. (2021) overcome the dialog entity inconsistency while achieving an advantageous computational footprint, rendering adapters particularly suitable for multi-domain specialization. However, QASL is the first example of the successful incorporation of adapters to the SL task, and also with an extra focus on the most challenging low-data scenarios.

7 Conclusion

We have demonstrated that reformulating slot labeling (SL) for dialog as a question answering (QA) task is a viable and effective approach to the SL task. Our comprehensive evaluations over two standard SL benchmarks have validated the effectiveness and robustness of the proposed QASL approach, yielding improvements over state-of-the-art SL models, especially in the most challenging, few-data setups. QASL is a very versatile framework, which can profit both from manually created and automatically created QA resources, and is applicable to an array of pretrained language models. Finally, we have shown how to efficiently fine-tune effective domain-specific SL models.

Limitations. Our current evaluation focuses only on slot labeling while earlier works show potential of QA-based intent detection. We have also not explored non-conversational domains.

References

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of EMNLP 2018*, pages 5016–5026.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45.

Guan-Lin Chao and Ian Lane. 2019. BERT-DST: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *Proceedings of Interspeech 2019*, pages 1468–1472.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. BERT for joint intent classification and slot filling. *CoRR*, abs/1902.10909.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *Proceedings of ICLR 2020*.

Samuel Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-ConveRT: Few-shot span extraction for dialog with pretrained conversational representations. In *Proceedings of ACL 2020*, pages 107–121.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13.

Shuyang Gao, Sanchit Agarwal, Di Jin, Tagyoung Chung, and Dilek Hakkani-Tur. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 79–89.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. 2019. Dialog state tracking: A neural reading comprehension approach. In *Proceedings of SIGDIAL 2019*, pages 264–273.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of ACL-IJCNLP 2021*, pages 3816–3830.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gasic. 2020. TripPy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of SIGDIAL 2020*, pages 35–44.

Matthew Henderson and Ivan Vulić. 2021. ConVEx: Data-efficient and few-shot slot labeling. In *Proceedings of NAACL-HLT 2021*, pages 3375–3389.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Proceedings of NeurIPS 2020*.

Yutai Hou, Wanxiang Che, Yongkui Lai, Zhihan Zhou, Yijia Liu, Han Liu, and Ting Liu. 2020. Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of ACL 2020*, pages 1381–1393.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of ICML 2019*, pages 2790–2799.

Chia-Chien Hung, Anne Lauscher, Simone Paolo Ponzetto, and Goran Glavaš. 2021. DS-TOD: efficient domain specialization for task oriented dialog. *CoRR*, abs/2110.08395.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR 2015*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proceedings of ICLR 2020*, volume abs/1909.11942.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *CoRR*, abs/2102.07033.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in Natural Language Processing. *CoRR*, abs/2107.13586.

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. *CoRR*, abs/1907.11692.

Zihan Liu, Genta Indra Winata, Peng Xu, and Pascale Fung. 2020. Coach: A coarse-to-fine approach for cross-domain slot filling. In *Proceedings of ACL 2020*, pages 19–25.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of COLING 2020*, pages 480–496.

Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. 2020a. Plug-and-play conversational models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2422–2433.

Andrea Madotto, Zihan Liu, Zhaojiang Lin, and Pascale Fung. 2020b. Language models as few-shot learner for task-oriented dialogue systems. *CoRR*, abs/2008.06239.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tür. 2020. DialoGLUE: A natural language understanding benchmark for task-oriented dialogue. *CoRR*, abs/2009.13570.

Shikib Mehri and Maxine Eskénazi. 2021. GenSF: Simultaneous adaptation of generative pre-trained models and slot filling. In *Proceedings of SIGDIAL 2021*, pages 489–498.

Mahdi Namazifar, Alexandros Papangelis, Gokhan Tur, and Dilek Hakkani-Tür. 2021. Language model is all you need: Natural language understanding as question answering. In *Proceedings of ICASSP 2021*, pages 7803–7807.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2021. Soloist: Buildingtask bots at scale with transfer learning and machine teaching. *Transactions of the Association for Computational Linguistics*, 9:807–824.

Jonas Pfeiffer, Aishwarya Kamath, Andreas RückÍe, Cho Kyunghyun, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of EACL 2021*.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for Natural Language Processing: A survey. *CoRR*, abs/2003.08271.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of ACL 2018*, pages 784–789.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of AAAI 2020*, pages 8689–8696.

Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. 2005. Let's go public! Taking a spoken dialog system to the real world. In *Ninth European conference on speech communication and technology*.

Evgeniia Razumovskaia, Goran Glavaš, Olga Majewska, Anna Korhonen, and Ivan Vulić. 2021. Crossing the conversational chasm: A primer on multilingual task-oriented dialogue systems. *CoRR*, abs/2104.08570.

Sebastian Ruder. 2021. Recent advances in language model fine-tuning.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

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 *Proceedings of NeurIPS 2017*, pages 5998–6008.

Weizhi Wang, Zhirui Zhang, Junliang Guo, Yinpei Dai, Boxing Chen, and Weihua Luo. 2021. Task-oriented dialogue system as natural language generation. *arXiv preprint arXiv:2108.13679*.

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

Steve Young. 2002. Talking to machines (statistically speaking). In *Seventh International Conference on Spoken Language Processing*.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *CoRR*, abs/2106.10199.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of ACL 2020: System Demonstrations*, pages 270–278.

Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2021. Retrospective reader for machine reading comprehension. In *Proceedings of AAAI 2021*, pages 14506–14514.

Li Zhou and Kevin Small. 2019. Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *CoRR*, abs/1911.06192.

A Statistics and Full Results on RESTAURANTS-8k and DSTC8

- Table 4 provides the exact number of examples over all slots for all the training data splits in RESTAURANTS-8k and DSTC8.
- Table 5 gives the exact scores related to Figure 3 in the main paper.
- Table 6 provides the exact scores related to Figure 4 in the main paper.

B Brief Analysis of DSTC8

Performance of QASL in the full-data scenarios already leaves little room for improvement on DSTC8 in future work. The most challenging slots are *pickup date* and *dropoff date* from the *Rental Cars* domain. As with RESTAURANTS-8k, we again observe that some mistakes made by the SL models can be attributed to ambiguous or wrong annotations. For example, we find 2 examples where a car is rented for a single day: whether the date is *pickup date* or a *dropoff date* is ambiguous.

	RESTAURANTS-8k	DSTC8			
		Buses	**Events**	**Rental Cars**	**Homes**
1/128	64	–	–	–	–
1/64	128	–	–	–	–
1/32	256	34	46	64	26
1/16	512	70	93	129	54
1/8	1024	141	187	258	109
1/4	2049	283	374	516	218
1/2	4099	566	749	1032	437
1	8198	1133	1498	2064	874
Test	3731	377	521	587	328

Table 4: Statistics of the data splits extracted from the RESTAURANTS-8k and DSTC8 datasets.

	GenSF	**ConVEx**	**QANLU**	**RoBERTa**	**RoBERTa-L**	**DistilBERT**
1/128	72.2	71.7	72.9	84.0	**84.5**	72.5
1/64	76.1	76.0	83.5	85.2	**87.2**	77.2
1/32	82.1	81.8	86.9	**89.9**	**89.9**	82.0
1/16	89.7	86.4	90.4	91.9	**92.0**	86.9
1/8	91.8	90.6	90.7	92.2	**92.9**	87.9
1/4	93.2	92.5	91.0	94.4	**94.6**	89.2
1/2	94.3	94.1	94.0	95.4	**95.6**	90.7
1	96.1	96.0	95.2	**96.1**	**96.1**	91.8

Table 5: Average F1 scores across all slots for the evaluation on the RESTAURANTS-8k test set.

	GenSF	**ConVEx**	**QASL**
Buses			
1/32		59.20	**92.80**
1/16		75.20	**93.30**
1/8		84.00	**95.50**
1/4	90.50	86.70	**95.70**
1/2		92.60	**96.10**
1	**98.10**	96.00	96.50
Events			
1/32		54.00	**76.20**
1/16		66.60	**89.10**
1/8		82.20	**92.70**
1/4	91.20	87.20	**95.80**
1/2		87.30	**97.60**
1	94.70	91.70	**97.80**
Rental cars			
1/32		50.3	**83.9**
1/16		60.6	**87.0**
1/8		77.6	**95.9**
1/4	93.70	87.4	**95.9**
1/2		91.7	**96.5**
1	**96.90**	92.0	96.3
Homes			
1/32		92.0	**95.4**
1/16		92.3	**95.0**
1/8		94.8	**97.9**
1/4	86.70	94.5	98.1
1/2		95.6	**98.7**
1	93.50	98.3	**99.1**

Table 6: Average F1 scores on the DSTC8 single-domain data sets.