Injecting Domain Knowledge in Language Models for Task-Oriented Dialogue Systems

*Denis Emelin[1], Daniele Bonadiman[2], *Sawsan Alqahtani[3,4], Yi Zhang[2], and Saab Mansour[2]

[1]University of Edinburgh , denis.emelin@gmail.com
[2]AWS AI Labs , {dbonadim,yizhngn,saabm}@amazon.com
[3]Princess Nourah Bint Abdulrahman , saalqhtani@pnu.edu.sa
[4]National Center of AI , sawalqahtani@nic.gov.sa

Abstract

Pre-trained language models (PLM) have advanced the state-of-the-art across NLP applications, but lack domain-specific knowledge that does not naturally occur in pre-training data. Previous studies augmented PLMs with symbolic knowledge for different downstream NLP tasks. However, knowledge bases (KBs) utilized in these studies are usually large-scale and static, in contrast to small, domain-specific, and modifiable knowledge bases that are prominent in real-world task-oriented dialogue (TOD) systems. In this paper, we showcase the advantages of injecting domain-specific knowledge prior to fine-tuning on TOD tasks. To this end, we utilize light-weight adapters that can be easily integrated with PLMs and serve as a repository for facts learned from different KBs. To measure the efficacy of proposed knowledge injection methods, we introduce *Knowledge Probing using Response Selection* (KPRS) – a probe designed specifically for TOD models. Experiments[1] on KPRS and the response generation task show improvements of knowledge injection with adapters over strong baselines.

1 Introduction

Pre-trained language models (PLMs), such as BERT (Devlin et al., 2018), BART (Lewis et al., 2020), GPT (Brown et al., 2020), and XLNet (Yang et al., 2019), have advanced the state-of-the-art of various natural language processing (NLP) technologies and demonstrated an exceptional ability to store and utilize linguistic, factual, and commonsense knowledge. Consequently, PLMs form the backbone of many recent NLP applications and have been successfully employed as modular components in the context of task-oriented dialogue (TOD), responsible for sub-tasks including dialogue state tracking and response generation (Hosseini-Asl et al., 2020; Lee et al., 2021).

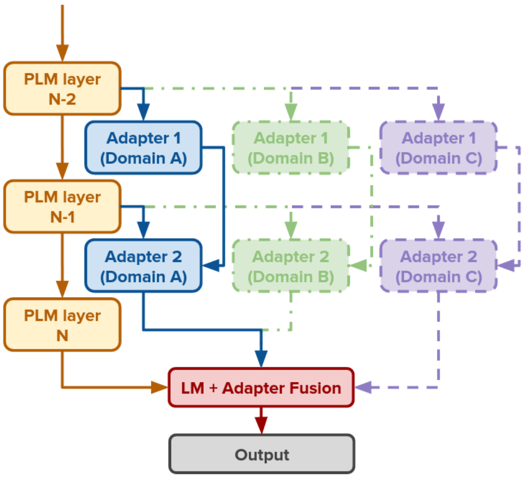


Figure 1: A high-level representation of the KB-adapter architecture (decoder only, for clarity). Adapter states are fused with the hidden states of the PLM to produce a knowledge-informed predictive distribution. Dashed elements are used only if multiple adapters are active.

Since they are exposed to large quantities of general data during training, PLMs store a wide variety of diverse and general knowledge in their parameters (Petroni et al., 2019) such as capitals of nations, biographical details of famous individuals, and other facts of varying granularity. Commercially deployed TOD systems, however, typically require access to more restricted, domain-specific categories of knowledge in order to produce informative and factually accurate responses to user queries.[2] Such information may include addresses of particular local attractions, detailed restaurant menus, train routes, or ticket prices, and is unlikely to be found in the PLM's training data. Due to its specialized nature, this knowledge is often stored in external knowledge bases (KBs) that are accessed at run-time by TOD systems via external queries.

* Work performed while at AWS AI Labs

[1] https://github.com/amazon-research/domain-knowledge-injection

[2] The term *domain* here refers to a specific application usecase (e.g. expedia.com (travel) and opentable.com (restaurant) represent different domains).

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 11962–11974
December 7-11, 2022 ©2022 Association for Computational Linguistics

This process introduces additional complexity into the dialogue model design and requires implementing KB queries and code wrappers as part of system setup, causing a substantial overhead especially for non-experts. Querying external KBs can also be disadvantageous when the KB is small, or is not changing in real time (as is the case with catalogs, restaurants' menus, etc). We identify the decoupling of domain-specific knowledge from the dialogue model as a shortcoming to be remedied and instead propose to inject this knowledge directly into the model's parameters. This eliminates the need for querying external KBs, streamlining the creation and deployment of TOD systems.

Injecting domain-specific information into TOD systems that can guide and inform model behavior and may be subsequently updated and modified by the user is not a trivial task. Ideally, this should be accomplished in a manner that is efficient, architecture-agnostic, and compatible with off-the-shelf PLMs. In order to satisfy these requirements, we adopt **light-weight adapter networks as repositories of domain-specific knowledge** (**KB-adapters** for short). Such adapters can be trained to memorize KB facts[3] and integrated into pretrained PLMs through the fusion of hidden representations, as illustrated in Figure 1. Our work is in line with past studies that demonstrated the utility of adapters as stores of factual and linguistic knowledge outside of TOD (Wang et al., 2020). Importantly, injecting knowledge into TOD models through adapters is computationally less demanding than injecting domain-specific facts by finetuning entire dialogue models on synthetic data, as explored in (Madotto et al., 2020), which facilitates efficient updating of the injected knowledge.

To quantify the success of the knowledge injection procedure, we develop the **K**nowledge **P**robing using **R**esponse **S**election (KPRS) task and benchmark (see §3). KPRS leverages contrastive dialogue response pairs to probe the extent of memorization of domain-specific facts by the evaluated dialogue model, whereby one response is consistent with the corresponding KB, while the other is not. To our knowledge, both KPRS and the use of adapters for domain-specific knowledge injection in TOD represent novel contributions of our work. We conduct experiments that evaluate PLMs equipped with domain-specific KB-adapters on the KPRS benchmark as well as the more conventional response generation (RG) task, comparing them against strong baselines.

Our contributions can be summarized as follows:

- We define and implement adapter-based methods for injecting highly specific and retrievable domain knowledge into TOD models
- We design and develop the KPRS probing task that can be used to evaluate the effectiveness of knowledge injection for TOD systems
- We show that PLMs with KB-adapters are usually preferable to knowledge-unaware and sequentially-finetuned PLMs for TOD

2 KB-Adapters for Domain-Specific Knowledge Injection

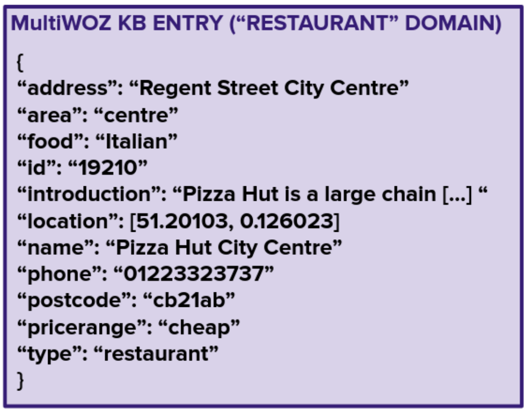
We conceptualize KB adapters as repositories of domain-specific information that guide the PLMs' predictions to be consistent with KB contents. The proposed knowledge injection process is divided into two stages: (1) **Memorization**: adapters are trained to memorize domain-specific KB facts; (2) **Utilization**: PLMs are trained to leverage adapters when reasoning about entities and their attributes.

During the memorization stage, adapters are connected to the frozen PLM and tasked with reconstructing corrupted KB facts, thereby memorizing associations between entity and attribute mentions. During the utilization stage, the PLM (now unfrozen) is given access to frozen adapters and learns to leverage their memorized knowledge to make more accurate predictions on downstream tasks such as RG. As a result, PLMs can generalize to unseen inputs by virtue of their domain-general pretraining while receiving domain-specific guidance in their predictions by the knowledge encoded in adapter representations.

When training KB-adapters, we allocate a single adapter for each individual domain KB (e.g. *hotel* or *restaurant*). This results in shorter training times per adapter and (if needed) facilitates efficient re-training of adapters to reflect changes in the associated KBs.[4] This allows for a straightforward extension of TOD systems equipped with KB-adapters to new domains, as this only requires training a single, new domain-specific adapter that can be used in concert with existing ones. Nevertheless, we also consider a setting where we train a

[3] We use the term *fact* to refer to individual KB entries.

[4] E.g. if the user updates the prices of certain items on a restaurant's menu.



MultiWOZ KB ENTRY ("RESTAURANT" DOMAIN)

```
{
"address": "Regent Street City Centre"
"area": "centre"
"food": "Italian"
"id": "19210"
"introduction": "Pizza Hut is a large chain [...] "
"location": [51.20103, 0.126023]
"name": "Pizza Hut City Centre"
"phone": "01223323737"
"postcode": "cb21ab"
"pricerange": "cheap"
"type": "restaurant"
}
```

Figure 2: Example MultiWOZ 2.2 KB entry.

single, mixed-domain adapter on the concatenation of all KBs in our experiments (see §5.5).

2.1 System Overview

Unlike the vast amounts of data used to pre-train PLMs, information stored in KBs is usually structured and does not resemble natural language expressions. Figure 2 shows a single KB entry (or *fact*) from the MultiWOZ 2.2 dataset (Budzianowski et al., 2018; Ye et al., 2021). Since KB-adapters need to be compatible with PLMs and their internal representations, we therefore convert KB entries prior to the memorization stage from their initial format into declarative statements of varying complexity (§2.2). Each statement mentions exactly one entity (e.g. a restaurant's name) and one or more entity attributes (e.g. the types of cuisine served by a restaurant). Each statement is subsequently corrupted by masking out a single attribute.[5] By denoising the input sequence, adapters learn to correlate entities with their attributes, effectively memorizing entire KBs with high accuracy (§2.3). The obtained KB-adapters are utilized to guide PLMs' predictions during fine-tuning on downstream TOD tasks (§2.4).

In our experiments, BART (Lewis et al., 2019) is chosen as the PLM that forms the backbone of the adapter-augmented TOD model, due to its competitive performance on generative tasks.[6] While the proposed knowledge injection approach is agnostic to the choice of particular PLM, we leave such validation for future work.

We employ bottleneck adapters (Houlsby et al., 2019) due to their established effectiveness and insert them after the final layer of the encoder and decoder.

atomic facts	Pizza Hut City Centre is located in the *centre* area of the city. Pizza Hut City Centre serves food in the *cheap* price range. The postcode of Pizza Hut City Centre is *cb21ab*. . . .
composite facts	Pizza Hut City Centre is a *restaurant* that serves *Italian* food in the *cheap* price range. It is located at *[51.20103, 0.126023]*, in the *centre* area of the city, in the *cb21ab* postcode. Its phone number is *01223323737*.

Table 1: Examples of the natural language formats used to represent KB facts in our study. Entity mentions are underlined, whereas entity attributes are *italicized*.

The PLM's hidden state given to the adapter as input is combined with the adapter's output using a weighted fusion function which is a linear transformation of the PLM's hidden state followed by a softmax activation that produces the fusion weights. This allows the final model to dynamically adjust the extent to which adapter knowledge is used at each prediction step. In this work, we ran two sets of experiments by applying this gating function to either the logits obtained from both the PLM and the adapters, or to their pre-logit hidden states.

We train a single encoder and a single decoder adapter per domain (hyper-parameter settings are reported in Appendix C).[7]

2.2 From KB Facts to Declarative Statements

Previous studies that investigated knowledge injection methods often use relational tuples to represent individual facts contained within a KB, e.g. where an entity is connected to one of its attributes via the relevant relation: `[Pizza Hut City Centre, food, Italian]`. While this knowledge representation format has been found to be effective in the past, our preliminary studies indicated that the mismatch between the natural language input format expected by a PLM and the structured tuple causes slight performance degradation. Hence, we choose to represent individual KB entries as natural language statements

[5] The entity mention is never masked out, as multiple entities can have the same attribute resulting in ambiguous model inputs, e.g. multiple restaurants can serve *Indian* food.

[6] We utilize the `BART-Large` provided as part of the `Transformers` library (Wolf et al., 2019).

[7] We also investigated several other fusion functions, including unweighted state averaging, state concatenation followed by a projection as used in (Wang et al., 2020), attention, GRU cell, and a combination of softmax distributions produced separately by the PLM and the adapter. However, neither of these performed better than the proposed approach.

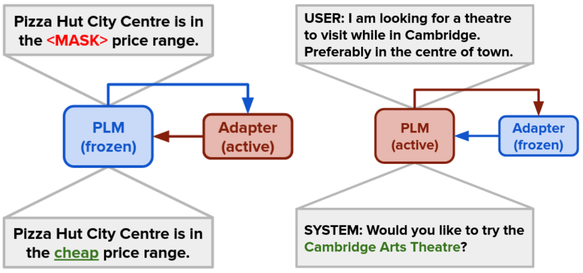

Pizza Hut City Centre is in the <MASK> price range.
PLM (frozen)
Adapter (active)
Pizza Hut City Centre is in the cheap price range.
USER: I am looking for a theatre to visit while in Cambridge. Preferably in the centre of town.
PLM (active)
Adapter (frozen)
SYSTEM: Would you like to try the Cambridge Arts Theatre?

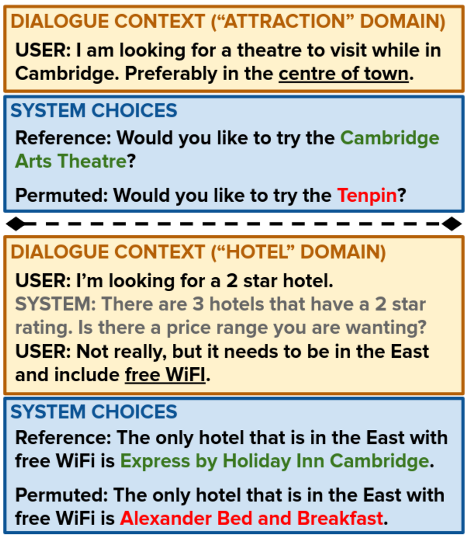
Figure 3: On the *left*, a schematic representation of the **memorization** stage, where the adapter is trained to memorize KB contents by reconstructing corrupted statements derived from KB facts. Note that the PLM parameters are frozen at this stage. On the *right*, a representation of the **utilization** stage, where the adapter-augmented PLM is fine-tuned on a downstream TOD task and learns how to utilize adapter knowledge. Note that the adapter parameters are frozen at this stage.

that are fully consistent with the data seen by the PLM during pretraining.

There are several intuitive ways in which a KB entries can be translated into natural language statements. Referring again to Figure 2, we consider (1) **atomic** statements, where each statement mentions the entity and one of its attributes, connected by the attribute's relation, and (2) **composite** statements where each statement communicates the entirety of the entry, covering all provided entity attributes and relations. Table 1 illustrates both formats based on the MultiWOZ KB entry in Figure 2. All statements are derived by filling-in pre-defined, human-authored templates with the appropriate entity and attribute values.[8] Designing the templates introduces minimal overhead, as they reuse attribute designations where possible and do not introduce any information beyond the contents of KB entries. The exhaustive list of templates used in our experiments is provided in Tables 9 and 10. During the memorization stage, KB-adapters are trained on a mixture of all atomic and composite facts, so as to familiarize the TOD model with different representations of the same information.

2.3 Memorization Stage

Following the construction of natural language representations of KB facts, the memorization stage involves training adapters to memorize and recall KB information. As shown on the *left* in Figure 3, the adapter-augmented PLM learns to reconstruct masked declarative statements that are derived from KB contents, whereby the weights of the PLM itself are kept frozen – only adapter parameters are being updated. By filling-in masked tokens, adapters learn correlations between entities (e.g. hotel names) and their attributes (e.g. phone numbers). Adapter training resembles masked language modeling and is easy to implement and scale.



DIALOGUE CONTEXT ("ATTRACTION" DOMAIN)

USER: I am looking for a theatre to visit while in Cambridge. Preferably in the centre of town.

SYSTEM CHOICES

Reference: Would you like to try the Cambridge Arts Theatre?

Permuted: Would you like to try the Tenpin?

DIALOGUE CONTEXT ("HOTEL" DOMAIN)

USER: I'm looking for a 2 star hotel.
SYSTEM: There are 3 hotels that have a 2 star rating. Is there a price range you are wanting?
USER: Not really, but it needs to be in the East and include free WiFI.

SYSTEM CHOICES

Reference: The only hotel that is in the East with free WiFi is Express by Holiday Inn Cambridge.

Permuted: The only hotel that is in the East with free WiFi is Alexander Bed and Breakfast.

Figure 4: Samples from the KPRS benchmark. Each sample consists of (1) a dialogue context that includes the available history and the active user turn and (2) two candidate responses to be scored by the model – a reference response that is consistent with both the dialogue context and the KB, and a perturbed response that is not. Reference values are set in **green** and perturbed values are set in **red**. Note that "Tenpin" is not in the centre area and "Alexander Bed and Breakfast" does not have free WiFi according to their respective KB entries.

2.4 Utilization Stage

After the memorization stage, PLMs are trained to leverage the domain-specific knowledge encoded in adapter representations with the goal of producing more accurate predictions on a downstream task, such as RG, as illustrated on the *right* in Figure 3. Throughout this fine-tuning process, adapter parameters are kept frozen so as to preserve the domain-specific knowledge injected during the memorization stage. PLM parameters, on the other hand, are unfrozen to allow the model to learn to exploit adapter representations.

[8] We note that we did not optimize the templates' design as part of our investigation. Our goal in creating the templates was to render structured KB content into natural language without introducing any superfluous information, so as to verify the efficacy of our adapter-based knowledge injection method without additional confounding factors.

3 Knowledge-Probing using Response Selection (KPRS) Benchmark

In this study, we investigate the ability of language models to verify and retrieve domain-specific facts within the TOD setting. To this end, we propose the "**K**nowledge-**P**robing using **R**esponse **S**election" (KPRS) task and the associated benchmark. KPRS allows us to examine whether domain-specific knowledge, such as entities and their attributes, that is stored within the parameters of the evaluated model can be successfully accessed and guide the model's predictions. Being knowledgeable about domain-specific entities in this manner can benefit dialogue models when reasoning about and replying to user queries. We show this to be the case for the response generation task in §5.3.

KPRS is a contrastive evaluation benchmark that measures whether the probed model has memorized and can accurately retrieve domain-specific knowledge contained within a specified KB. It is derived from MultiWOZ 2.2 dialogues (Zang et al., 2020) (development and test portions only) and covers four domains: *restaurant*, *hotel*, *attraction*, and *train*. Given a dialogue context, the task presented to the evaluated model is to score responses that are either compatible or incompatible with the information contained in the KB.

Importantly, KPRS should not be regarded as a stand-alone evaluation task, but rather as a probing mechanism that can offer informative insights into a model's ability to access domain-specific facts stored within its parameters, similar to other knowledge probes, e.g. (Petroni et al., 2019). Specifically, a fact-aware model should be able to distinguish between an **appropriate ("reference") dialogue response** that is compatible with the knowledge base information from an **inappropriate ("distractor") response** that contradicts the domain-specific knowledge. By design, the two responses are minimally different – identical except for attribute values associated with entities described in the KB, such as restaurant names or departure times of trains. Hence, to identify the correct dialogue response, a model must be able to distinguish values that are compatible with domain-specific information from those that are not.

3.1 Benchmark Design

In order to derive KPRS from MultiWOZ 2.2 development and test set dialogues, we (1) extract dialogue contexts that precede a system response that contains a mention of an entity from the KB or its attributes, and (2) perturb the corresponding system response to make it incompatible with the KB by modifying said entity and attribute mentions.

Different perturbation strategies are used for different types of attribute slots. For phone numbers, a single digit is randomly changed. For integers (e.g. denoting the price of a train ticket), we randomly increment or decrement the numbers by a small amount. For other slot types, distractor values are chosen so that they differ from the reference value while producing inadmissible responses. Distractors are chosen adversarially, i.e., candidates are sampled from the KB until the perturbed response becomes incompatible with the domain-knowledge and the dialogue context up to the response, while also achieving a lower sentence-level perplexity than the reference response according to a filter-LM (`BART-Large`). The latter is to ensure the well-formedness and plausibility of the perturbed responses. To guarantee that the perturbed response is indeed unsuitable, we make sure that the selected distractor does not share attriutes that have been mentioned in the dialogue context with the replaced slot value.[9]

Figure 4 shows examples included in the KPRS benchmark. Each KPRS sample contains the **dialogue context** that includes reference dialogue states, and two response options – **reference response** and **distractor response**. Overall, the KPRS benchmark dataset includes 3,055 samples (1,711 single-domain, 1,324 multi-domain). Samples had been derived from 831 unique dialogues / 1,997 unique dialogue contexts. On average, 3.65 samples were obtained from each individual dialogue / 1.52 samples from each individual dialogue context.

4 Experimental Setup

4.1 Knowledge Base Resource

Throughout our experiments, we use MultiWOZ 2.2 (Zang et al., 2020) which contains several relatively small-scale domain-specific KBs that are aligned with task-oriented dialogues.[10] After fil-

[9] E.g. if the response originally mentioned the name of a restaurant that serves Italian food and the dialogue context up to the response only mentions Italian cuisine as a desired restaurant property, the distractor is explicitly chosen, using string-matching heuristics, to be a restaurant that serves some other type of food, so as not to unintentionally yield a valid response.

[10] In practical settings, businesses maintain similar knowledge bases in-house which could be utilized in TOD servicees.

restaurant	hotel	attraction	train
1,540	594	1,106	39,592

Table 2: Number of facts in each KB.

restaurant	hotel	attraction	train
98.1	98.2	97.6	93.2

Table 3: Fact memorization accuracy for KB-adapters.

tering out KBs with missing information, we are left with four domains: *restaurant*, *hotel*, *attraction*, and *train*. Table 2 shows the number of facts available per domain. Note the substantial gap in the number of facts where *trains* is approximately 25X to 66X larger than the other domains.

4.2 Intrinsic Evaluation

To examine whether they can accurately retrieve the injected KB facts, we task knowledge-augmented PLMs with reconstructing masked facts, using inputs of the same format as described in §2.2. Since this task measures success as a model's ability to memorize and recall learned KB information rather than generalize it to unseen inputs, we evaluate our models on the same set of data as was used for knowledge injection as part of the memorization stage. Memorization accuracy is employed as the evaluation metric, representing the number of facts that have been correctly reconstructed. We refer to this task as *fact memorization* task.

4.3 Downstream Evaluation

Additionally, we evaluate our models on the KPRS probe (§3) as well as the response generation (RG) task. While KPRS directly estimates **models' preference** for dialogue continuations that are either consistent or inconsistent with KB information, RG examines **model's ability to integrate the injected KB knowledge** into the generated response as part of the TOD pipeline.

For KPRS, we fine-tune BART-large on the training data for each domain, using correct responses as targets, and evaluate subsequent model performance on the KPRS benchmark. An augmented PLM that can accurately access the injected domain-specific facts is expected to assign a higher likelihood to the reference response, compared to the permuted distractor. Response selection accuracy is used as the evaluation metric, defined as (c/N), where N is the total number of contrastive sentence pairs and c is the number of pairs in which the reference response (i.e. the one consistent with the KB) is assigned lower perplexity by the model.

For RG, given a dialogue context, models must generate a response that is consistent with KB facts without performing external KB queries. To test the model's ability for fact retrieval, we use unweighted mean of two informative metrics: inform rate (n/N) and success rate (m/N) (Zang et al., 2020), where N is the total number of turns in the test set, n is the number of turns in which the entities generated by the model are all consistent with the KB, and m is the number of turns in which the model generation provides at least as much of the user-requested information as the gold response.[11]

4.4 Baselines

We compare the performance of the knowledge-injected model with two baselines: (1) BART-large without any knowledge augmentation; (2) BART-large that has been sequentially fine-tuned on each KB (Seq-BART). We fine-tune all models on the downstream task prior to the downstream evaluation.

5 Results & Analysis

We examine the models' ability to memorize and retrieve facts learned from the knowledge base in §5.1 and the impact of knowledge injection on downstream tasks in §5.2 and §5.3. Models were evaluated in the single-domain setting where only one single adapter corresponding to the specified domain was active at evaluation time with test samples belonging exclusively to the adapter domain (a multi-domain setting is discussed in §5.5)

5.1 Fact Memorization

As discussed in §4.2, we evaluate whether the knowledge-augmented model is able to successfully denoise masked facts seen during training, thus testing its memorization capabilities. Table 3 shows the results of the fact memorization task for BART equipped with KB-adapters. The memorization accuracy is generally very high across all domains and appears to correlate with KB size.

[11] BLEU (Papineni et al., 2002), as typically used for text generation, is not sufficient as an evaluation metric for our purpose. Previous work evaluated generated responses that contain slot value placeholders instead of concrete information such as entity attributes, as in the case in our study. In addition, any evaluation of response factuality must consider all permissible entities given the dialogue context, rather than only one out of many, as is implicitly done by BLEU.

Model	rest.	hotel	attr.	train	all
BART	70.8	72.5	71.3	78.9	76.5
Seq-BART	71.5	72.1	72.7	74.4	75.6
ada-logits	**81.5**	**83.1**	**81.2**	**94.3**	78.2
ada-hidden	81.3	82.0	80.6	94.0	**78.4**

Table 4: Response selection accuracy on KPRS. *ada-logits* and *ada-hidden* refer to experiments utilizing KB-adapters with different fusion mechanisms (either at the level of logits or pre-logits hidden states).

5.2 Knowledge-Probing using Response Selection (KPRS)

Table 4 reports the performance of the knowledge-augmented PLM compared to baselines introduced in §4.4. We found that injecting domain-specific knowledge into the PLM significantly improves KPRS accuracy – by 9-15% – compared to BART. The largest improvement can be observed in the *train* domain, which is at odds with the fact memorization results (§5.1), where our model underperformed on that domain. As such, while perfect memorization of a of all facts contained within a large KBs remains a challenge in the current training setup, the domain knowledge embedded within the adapter network can nevertheless be effectively exploited by the PLM.

5.3 Response Generation (RG)

Presumably, having access to the domain knowledge stored in KB-adapters should enable a PLM to generate responses that are more consistent with the respective KBs. Table 5 reports the results for our RG experiments, providing empirical support for this hypothesis. Interestingly, a large discrepancy can be observed for the *hotel* domain between the two examined representation fusion techniques (*ada-logit* that combines PLM and adapter representations at the logit level vs. *ada-hidden* that combines their pre-logits hidden states). We hypothesise that this is, at least in part, due to the *hotel* KB containing a small number of facts, which may have caused instability during training. Accordingly, although knowledge injection can clearly benefit generation of factual system responses in both the single-domain setting, the extent of the improvements is contingent on the target domain and its properties, as is the best-performing representation combination function.[12]

Model	rest.	hotel	attr.	train	all
BART	**54.7**	44.3	50.3	38.2	54.2
ada-logit	46.0	12.6	**69.7**	**55.0**	61.4
ada-hidden	53.3	**55.9**	68.6	48.6	**62.3**

Table 5: RG performance calculated as the average of inform rate and success rate metrics. The *all* column reports results for the multi-domain setting.

Model	rest.	hotel	attr.	train	all
BART	12.59	11.53	15	17.71	13.41
ada-logit	**12.69**	6.5	15.09	**19.33**	**15.98**
ada-hidden	10.39	**12.64**	**15.94**	17.56	15.33

Table 6: Response generation BLEU score performance.

Table 6 provides estimates of RG quality according to BLEU. Overall, we see minor to substantial improvements with respect to the BLEU metric over the baseline lacking KB-adapters. This can be taken as further evidence in support of the effectiveness of the proposed knowledge injection methodology. However, it should be noted that the extent of the observed improvements varies across domains and representation combination functions.

5.4 Randomly-initialized Adapters

We investigate how equipping PLMs with our proposed KB-adapters compares to equipping them with randomly-initialized adapters during the fine-tuning stage (a setting to which we refer as *rand-BART*). This effectively isolates the impact of knowledge injection on the KPRS and RG performance, by factoring out the increased model capacity due to the additional parameters introduced by the adapters. Table 7 shows the experimental results for both tasks. We find that injecting domain-specific knowledge into the PLM does indeed significantly improve KPRS performance – by 6-15% – compared with *rand-BART*, thus further validating our approach.

5.5 Integration of Multiple Knowledge Bases

The modular nature of of the proposed knowledge-injection method allows us to equip PLMs with multiple adapters, with each adapter encoding information from a different domain. This enables the augmented PLM to access facts from different domains simultaneously, without running the risk of catastrophic forgetting, whereby information from one domain overwrites previously acquired domain-specific knowledge, e.g. as a result of sequential fine-tuning. Aligned with our motivation to allow users to easily add and modify

[12] It would be valuable to investigate the general impact of KBs' size on the PLMs' performance. However, this falls outside the scope of this paper, as such study would require a greater diversity in the sizes of available KBs.

Model	rest.	hotel	attr.	train
KPRS				
rand-BART	70.3	76.6	74.4	79.6
ada-logits	**81.5**	**83.1**	**81.2**	**94.3**
ada-hidden	81.3	82.0	80.6	94.0
RG				
rand-BART	0	47.0	66.1	28.2
ada-logit	46.0	12.6	**69.7**	**55.0**
ada-hidden	**53.3**	**55.9**	68.6	48.6

Table 7: Response selection accuracy on KPRS and the average of inform and success rate metrics for RG. For RG in the the *restaurant* domain, rand-BART failed to converge given our hyper-parameter settings.

KBs in practical settings, we investigate whether our proposed system can effectively integrate information from multiple adapters. We utilize the same representation combination functions as described in §2.1, generalizing them to an unbound number of adapters by computing normalized fusion weights for each adapter and the PLM itself. In this *multi-domain* setting, multiple adapters are active simultaneously, while test samples are drawn from all four studied domains.

Tables 4 and 5 report multi-domain results for KPRS and RG in the *multi* column. For both tasks, we observe clear improvements compared to baseline models when providing the model with access to all domain-specific adapters simultaneously. However, we note that the gap between the adapter-augmented PLM and the best-performing baseline is much smaller compared to single-domain experiments where the model only has access to a single, relevant adapter (1.9% vs. 12.25% on average for KPRS and 8.1% vs. 11.6% on average for RG).

One reason for the limited improvements observed in the multi-domain setting could be the PLM's inability to correctly identify adapters corresponding to the dialogues' domains and to promote their representations. The more pronounced gains observed in the single-domain setting – where the model does not have to chose between multiple adapters – appears to support this interpretation. To verify our hypothesis, we preclude the need for adapter selection by instead training a single adapter on the concatenation of facts from all four domains, which preserves the multi-domain setting. Evaluating the performance of the resultant model on KPRS, we observe improvements over the multiple adapters setting, with *ada-logis* obtaining an accuracy of 83.0% and *ada-hidden* reaching 85.9%, thus improving over the best-performing baseline by a substantial **9.4%**. This, however, comes at the expense of increased training time during the memorization stage and a significant reduction in flexibility for the addition of new KBs (which will require costly re-training the single, multi-domain adapter rather than simply introducing a new single-domain adapter).

It may be possible to improve the performance of PLMs equipped with multiple single-domain adapters by implementing more expressive combination representation functions or by adjusting the training regime. We regard as a promising research direction that could more effectively extend the flexibility of adapter-based knowledge injection to more complex dialogue settings.

6 Related Work

6.1 Knowledge Injection

Our work contributes to the growing body of research that explores strategies for introducing external knowledge into the internal reasoning processes of PLMs, with the aim of aligning their predictions with respective knowledge sources (Colon-Hernandez et al., 2021). Previous work in this area incorporated linguistic (Lauscher et al., 2019; Wang et al., 2020), factual (Wang et al., 2020; Agarwal et al., 2020), and commonsense (Lauscher et al., 2020) knowledge into pretrained models, with studies differing in the exact format of the injected knowledge and potential modifications to the PLMs' architecture. Nevertheless, injection of highly specific, fine-grained, tabular information commonly associated with TOD (as exemplified by MultiWOZ 2.2 KBs) has so far received limited attention, both within dialogue literature and beyond. The use of natural language statements as the primary mechanism for injecting external information into PLMs has been previously considered in works such as (Lu et al., 2021), who trained a generative model to transform knowledge triplets into declarative statements. We rely on template-based generation, instead, to account for the relatively small size of our KBs, the highly structured nature of KB entries, and the lack of natural language sequences that can be trivially aligned with KB contents.

6.2 Knowledge-Grounded Dialogue

Of particular relevance to our work is the study by (Madotto et al., 2020) who fine-tune all parameters of a PLM on synthetic dialogues constructed so as to communicate all information contained within a TOD KB. The limitations of their approach, as noted by its authors, are that the synthetic dialogues are noisy and any subsequent updates to the injected KB information require finetuning the entire model anew which is computationally demanding. We address both issues by relying on grammatically sound templates during knowledge injection and by leveraging light-weight adapters that can be updated for a small fraction of cost incurred by updating the full PLM. The Adapter-Bot introduced in (Lin et al., 2021) is likewise related to our models in that it employs adapters in the context of TOD. However, rather than training adapters to memorize KB content that can be exploited by the dialogue model without additional supervision, the authors rely on knowledge-aligned dialogues to introduce domain-specific information into their model which may not always be available. More recently, (Fan et al., 2021) proposed equipping transformer models with specialized modules that fetch embedded information from external resources, integrating it into the model's reasoning process. While the authors apply their model to dialogue generation, their work differs substantially from ours, as they do not consider the task-oriented setting or structured KBs (instead using training set utterances and Wikipedia excerpts). However, combining knowledge memorization and differential information retrieval is a promising direction for future research.

Moreover, external knowledge has found application in dialogue literature outside of directly guiding response generation. For instance, (Lertvittayakumjorn et al., 2021) annotated dialogue data with constraint violations based on valid links between entities as specified in the corresponding KBs. Similar to KPRS, detection of constraint violations can be regarded as a probing task that provides insights about the ability of a dialogue model to reason about KB entities.

7 Limitations

One of the main limitations of the presented approach is its reliance on manually constructed fact templates. We experimented with fine-tuning KG-adapters directly on $< ent1, rel, ent2 >$ KB triples, but found that the use of templates improves the ability of models to apply the memorized knowledge in downstream applications. In light of this, possible future extensions of our work may include creation of domain-agnostic strategies for knowledge injection that do not necessitate manual design of templates for each new domain.

Another limitation comes from the fact that the proposed approach is suitable only for static and pseudo-dynamic KBs , i.e. that can change periodically, such as a seasonal menu or a database of cars manufactured by a company. However, it is not suited for real-time databases (e.g. databases that store the availability of rooms in a hotel) since for every KB change the corresponding adapter needs to be retrained in order to be updated.

Additionally, while injecting knowledge into the language model has been shown to be effective for making it available during fine-tuning on downstream tasks, the knowledge stored in the adapters' parameters might not be accurate enough for certain real world applications due to the imperfect fact memorization we observed in our experiments.

Finally, the introduced KPRS task only evaluates the extent to which a model can access factual information stored in its parameters. It does not not assess the model's ability to understand and use this knowledge for complex reasoning tasks, e.g. counting the number of cars in a specific price range, or listing the items on a menu that do not contain a certain ingredient. This could be an exciting direction for future research.

8 Discussion and Conclusion

In this study, we proposed a method for tightly integrating external knowledge with the internal representations of PLMs by storing domain-specific information within light-weight adapter networks that guide model predictions. Such adapters can memorize KB contents with high accuracy, which decreases slightly for larger KBs. An important contribution of our work is the KPRS probe designed to measure the ability of TOD models to reason about KB entities and their attributes. As part of our experiments, we showed that KB-adapters clearly benefit the identification and generation of TOD responses that are consistent with dialogue history and relevant KB entries, and showcased the advantages of using adapters for knowledge injection as opposed to sequential fine-tuning.

Our investigation demonstrates that dialogue models can access domain-specific knowledge without having to query external KBs. This is an important finding as it can pave the way towards reducing the query engineering overhead in chatbot design, thus lowering the entry barrier for developing and deploying real-world TOD systems.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2020. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.

Pedro Colon-Hernandez, Catherine Havasi, Jason Alonso, Matthew Huggins, and Cynthia Breazeal. 2021. Combining pre-trained language models and structured knowledge. *arXiv preprint arXiv:2101.12294*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. Augmenting transformers with knn-based composite memory for dialog. *Transactions of the Association for Computational Linguistics*, 9:82–99.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

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 *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Anne Lauscher, Olga Majewska, Leonardo FR Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavaš. 2020. Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers. *arXiv preprint arXiv:2005.11787*.

Anne Lauscher, Ivan Vulić, Edoardo Maria Ponti, Anna Korhonen, and Goran Glavaš. 2019. Informing unsupervised pretraining with external linguistic knowledge.

Chia-Hsuan Lee, Hao Cheng, and Mari Ostendorf. 2021. Dialogue state tracking with a language model using schema-driven prompting. *arXiv preprint arXiv:2109.07506*.

Piyawat Lertvittayakumjorn, Daniele Bonadiman, and Saab Mansour. 2021. Knowledge-driven slot constraints for goal-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3407–3419.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Zhaojiang Lin, Andrea Madotto, Yejin Bang, and Pascale Fung. 2021. The adapter-bot: All-in-one controllable conversational model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 16081–16083.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yinquan Lu, Haonan Lu, Guirong Fu, and Qun Liu. 2021. Kelm: Knowledge enhanced pre-trained language representations with message passing on hierarchical relational graphs. *arXiv preprint arXiv:2109.04223*.

Andrea Madotto, Samuel Cahyawijaya, Genta Indra Winata, Yan Xu, Zihan Liu, Zhaojiang Lin, and Pascale Fung. 2020. Learning knowledge bases with parameters for task-oriented dialogue systems. *arXiv preprint arXiv:2009.13656*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2021. Multiwoz 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. *arXiv preprint arXiv:2104.00773*.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. Multiwoz 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. *arXiv preprint arXiv:2007.12720*.

A Atomic vs. Compositional Fact Formats

While developing the memorization stage of the knowledge injection process, we compared the relative utility of representing KB facts as either atomic or compositional statements, as measured by the memorization accuracy attained by the adapter-augmented PLM. The results of this pilot experiment are summarized in Table 8, which paints a mixed picture. While atomic statements result in stronger memorization for the *restaurant*, *hotel*, and *attraction* domains, compositional statements are substantially more effective in the *trains* domain. We therefore decided to combine both formats for our main set of experiments, as the resultant mixture shows reasonable performance across all domains. Furthermore, exposing the model to different surface forms of the same underlying information is expected to enable better generalization for downstream tasks.

Model	rest	hotel	attract	train
atomic	**98.2**	**99.3**	96.7	88.7
composite	95.8	97.6	93.7	**97.0**
both	98.1	98.2	**97.6**	93.2

Table 8: Memorization accuracy when training adapters on different formats of declarative statements. *both* denotes the combination of atomic and compositional statements. Scores set in **bold** are the highest in their respective column.

B Ethical Considerations

Injection of external knowledge into dialogue models may have both ethical and legal implication, if said knowledge contains personal identifiable information (PII), such as social security numbers of addresses of private individuals. Such information would be memorized by the adapter-augmented model and potentially exposed during response generation, if there are no additional safeguards in place to prevent this scenario. For this reason, it is crucial to curate the memorized KBs by removing any and all instances of PII prior to the memorization stage.

C Hyper-parameters

All models were trained on V100 GPUs, using the PyTorch implementation of the BART-Large model distributed as part of the HuggingFace Transformers library (Wolf et al., 2019). The training loop employed the AdamW (Loshchilov and Hutter, 2017) optimizer. By conducting a grid search, we empirically determined that a learning rate (LR) of $3e^{-5}$ worked best for fine-tuning RG models and LR of $1e^{-6}$ yielded best results for KPRS. For knowledge injection, LR of $3e^{-5}$ was found to be effective. In all cases, LRs were kept constant across all domains. For all domains and experiments, we re-use the same bottleneck adapter configuration, by setting the size of the hidden layer to 769. All models were trained until convergence by terminating training after 10 epochs during which no improvement had been observed on the development set.

D Fact Templates

This section provides a complete, exhaustive list of all templates used in the generation of declarative statements derived from the MultiWOZ 2.2 KB facts.

Domain	Fact Type	Templates
restaurant	address	The restaurant {} is located at {}.
	area	The restaurant {} is located in the {} area of the city.’
	food	The restaurant {} serves {} food.
	phone	The phone number of the restaurant {} is {}.
	postcode	The postcode of the restaurant {} is {}.
	pricerange	The restaurant {} is in the {} price range.
	type	{} is a {}.
hotel	address	The hotel {} is located at {}.
	area	The hotel {} is located in the {} area of the city.
	internet	The hotel {} does{}have free internet.
	parking	The hotel {} does{}have free parking.
	phone	The phone number of the hotel {} is {}.
	pricerange	The hotel {} is in the {} price range.
	stars	The hotel {} is rated as {} stars.
	type	The hotel {} is a {}.
attraction	address	The attraction {} is located at {}.
	area	The attraction {} is located in the {} area of the city.
	entrance fee	The entrance fee for the attraction {} is {}.
	phone	The phone number of the attraction {} is {}.
	postcode	The postcode of the attraction {} is {}.
	pricerange	The attraction {} is in the {} price range.
	type	The attraction {} is {}.
train	arriveBy	The {} train arrives at its destination by {}.
	day	The {} train operates every {}.
	departure	The {} train departs from {}.
	destination	The destination of the {} train is {}.
	duration	The duration of the journey with the {} train is {}.
	leaveAt	The {} train leaves at {}.
	price	The ticket price for the {} train is {}.

Table 9: An exhaustive list of human-authored templates used to generate **atomic statements** for use in the memorization stage. Note that each domain is allocated exactly one template per entity attribute. Also note that the mask in *does{}have* allows for negation in cases where the attribute is negative (e.g. if a hotel does not have free WiFi).

Domain	Templates
restaurant	{} is a {} that serves {} food in the {} price range. It is located at {}, in the {} area of the city, in the {} postcode. Its phone number is {}.
hotel	The hotel {} is a {} in the {} price range. It is rated {} stars. It is located at {}, in the {} area of the city, in the {} postcode. Its phone number is {}. It does{}have free parking and it does{}have free internet.
attraction	The attraction {} is {} in the {} price range. The entrance fee is {}. It is located at {}, in the {} area of the city, in the {} postcode. Its phone number is {}.
train	The {} train departs from {} every {}. It leaves at {}. Its destination is {} where it arrives at {}. The duration of the journey is {}. The ticket price for this train is {}.

Table 10: An exhaustive list of human-authored templates used to generate **composite statements** for use in the memorization stage.