Task-Oriented Dialogue State Tracking with GPT-2

MPhil MLMI

MLMI 8 Neural Machine Translation

Candidate Number
J902G

Date:
April 17, 2022

1 Overview

The task of dialogue state tracking (DST) is an important one in spoken language processing and allows for two-way communication between a human user and a machine interface to assist the user with basic tasks such as information retrieval.

Historically DST models have mostly been modularised into 3 major sections: natural language understanding (NLU), dialogue management and state tracking, and natural language generation (NLG). This project focuses on the first two of these, specifically looking at ways of improving natural language understanding and belief state tracking.

The approach taken for NLU uses a GPT-2 model that has been trained on an extensive and domain heterogenous data-set. GPT-2 models have shown impressive performance on a large number of language tasks across a wide range of non-specialist domains due to the vast and varied data that they are trained on [2]. However, their performance is less good on specialised domains such as those considered in this practical, introducing the requirement for fine-tuning. Therefore the first part of this practical considers the fine-tuning of Huggingface's GPT-2 transformer[1] on a training set of the Multi-domain Wizard Of Oz data-set (MultiWOZ) version 2.1. This data-set has around 8,438 annotated human-to-human conversations that cover a wide range of topics relevant to task-oriented DST [1].

After fine-tuning, the desire is that there is now a NLU system capable of accurately transforming spoken sentences into slot-value pairs that form a belief state on a turn of a conversation. This belief state represents important task-specific facts that the system believes has been given, such as the type and price-range of a hotel to book when the task is about finding accommodation. For the purpose of this practical, these belief states are linearised into strings rather than hierarchical, which makes them easier to process by transformer based systems.

The second part of this practical then investigates the extension from the turn-level belief state tracking to a conversation-level dialogue-state tracking system. The advantage of this approach over turn-level belief state tracking is that each turn will not contain all of the information about a user request due to the natural way in which conversations flow, but over multiple turns the accumulation of information will likely contain a much better picture. For example, a user might ask "I want to book a cheap hotel for Friday", which gives information on the price-range and day, but not on the location. If the next user spoken line was "it needs to be in Cambridge", then the turn-level belief state captures the location but not the price-range or day. In this way individual turns hold insufficient information, whilst the conversation-level information over all turns is a lot richer and thus would likely result in much higher accuracy when benchmarked against the human annotated test-sets. This is the basis of DST, and the hypothesis that dialogue-level systems present a major improvement in conversation level average joint accuracy compared turn-level systems is investigated in the later half of this report.

[1] As distributed on https://huggingface.co/gpt2

2 Exercise GPT2DST.1

In part 1 of this practical, continued training was carried out on the partially fine-tuned GPT-2 model provided, with fine-tuning carried on from the batch 60,000 checkpoint. Every 20,000 batches the development loss was reported, with training loss reported at the end of every epoch (every 54,971 batches) also. The results of this training are shown in Table 1 below. The loss is measured using negative log likelihood, and so lower loss represents a higher output likelihood and hence better model performance.

Batch	Training Loss	Dev. Loss
60,000	-	0.2093
80,000	-	0.2323
100,000	-	0.2031
114,971	0.2325	0.2112
120,000	-	0.2254
140,000	-	0.2088
160,000	-	0.2284
169,942	0.1916	0.2104
180,000	-	0.2454
200,000	-	0.2866
220,000	-	0.2417
224,913	0.1566	0.2504
240,000	-	0.2754
260,000	-	0.2919
279,884	0.1216	0.3017

Table 1: Training and development set loss for the fine-tuning of the GPT-2 pre-trained model on the MultiWOZ human-to-human conversation texts. Continued training was used for this fine-tuning, using the provided partially fine-tuned model up to batch 60,000 as a checkpoint.

It is evident from the table above that, whilst the training set loss continues to decrease consistently with fine-tuning, the development set loss starts increasing after about batch 100,000, indicating that overfitting is likely happening on the MultiWoz training set.

Between batches 60,000 to around 110,000, the differences in results between Table 1 and Figure 3 from the practical hand-out are small and the trend is the same (decrease in training loss with gradual increase in development set loss). The discrepancies are likely due to the model above being trained via continued training from a checkpoint whereas the model in the handout is trained from scratch.

The fine-tuning process was executed using a SLURM job on the HPC containing the command:

```
python $BDIR/train-gpt2-dst.py $BDIR/checkpoints/nlu_flat_start/model.60000/
--train_data $BDIR/data_preparation/data/multiwoz21/processed/train/version_1/data.json
--dev_data $BDIR/data_preparation/data/multiwoz21/processed/dev/version_1/data.json
--args $BDIR/train_arguments.nlu.mwoz21.yaml -vv
>> slurm_logs/train-log.$JOBID 2> slurm_logs/train-err.$JOBID
```

Where `$JOBID` is the ID of the submitted SLURM job.

3 Exercise GPT2DST.2

Now that the GPT-2 transformer model has been fine-tuned on the MultiWOZ data-set to improve its performance in task-specialist domains, the next stage to building the NLU system is to decode the human spoken phrases into turn-level belief states. The input is of the format:

```
"example_id":  "PMUL0025-0",
"dst_input":  "<USR> can i get a good place for afternoon tea in the westside ?  "
```

Where the integer after the `-` in the `example_id` indicates the turn number in the conversation. The aim is to generate a belief state for this query, with the human-annotated target for this example being:

```
"restaurant food afternoon tea"
```

The generation of hypothesis belief states from the human requests is done via the provided decoding script `$BDIR/gpt2-dst.py`, which takes the fine-tuned model as input and uses it to convert the request to a preset format of linearised slot-value pairs. For example, in the example above the fine-tuned model would (ideally) recognise that the spoken phrase relates to the slot `restaurant food` and then find its associated value, which is `afternoon tea`. The output annotations for the predicted belief state are concatenated to the linearised user-to-system conversation in order to be more easily passed into the transfomer model. In this example, for the batch 80,000 checkpoint model, the predicted belief state was:

```
"<USR> can i get a good place for afternoon tea in the westside?
<bos>restaurant area west<eos>"
```

Although not the same as the human annotated target, this output is not a completely bad result, as it does capture correct information from the query which is missed in the human annotated version (in fact, the human annotated version takes an additional extra two turns before storing the area as west).

Scoring of the predicted belief states is executed via `$BDIR/multiwoz_dst_score.py`, which performs *turn-level* scoring of the hypotheses relative to the human annotated targets. This means that each turn's hypothesis belief state is treated individually and no net information retrieval over a whole conversation is considered. The metrics to judge performance are the average slot accuracy and the average joint accuracy, which measure on average how many of the slots are correct and how many of the combinations of slots and values are correct respectively. For this practical we focus on average joint accuracy only, as it contains information about both slots and values.

This decoding was executed for the fine-tuned GPT-2 model at batch checkpoints 80,000 through to 200,000. The average joint accuracy is shown below in Table 2. (Recall also that the fine-tuned model was trained via continual training from the checkpoint batch 60,000.)

Batch	Av. Joint Accuracy (%)
80,000	70.14
100,000	68.59
120,000	69.84
140,000	70.28
160,000	71.20
180,000	71.85
200,000	72.58

Table 2: Table of average joint accuracies for the turn-level belief states produced by the fine-tuned GPT-2 NLU system after a variable number of fine-tuning batches. The targets here are the human-annotated turn-level belief states.

The results in Table 2 show that increasing the number of batches that the GPT-2 model is fine-tuned on in general increases the average joint accuracy. This is somewhat surprising given the decrease in development set loss due to overfitting after a large number of batches in part 1. Nevertheless, the final average joint accuracy after 200,000 batches is a significant improvement over performance before any major fine-tuning had been carried out, as the practical hand-out reports the average joint accuracy after only 5,000 batches of fine-tuning to be 66.30% on the test set.

4 Exercise GPT2DST.3

With a NLU system that can convert dialogue into turn-level belief states, the next stage to generating a DST is to convert the turn-level beliefs into dialogue-level beliefs. This part focuses on the most simple means of doing this, which is to concatenate all of the turn-level belief states within each conversation to produce a running history of all hypotheses throughout the conversation.

This simple concatenation for each conversation is carried out using the provided `$BDIR/cc-dst.py` script. An example concatenation is as follows for a test conversation after 200,000 fine-tuning batches of the model:

```
"example_id": "PMUL0025-0",
"predicted_belief_state": "<USR> can i get a good place for afternoon tea in the westside? <bos>
restaurant area west <SEP> restaurant food afternoon tea <eos> ",


"example_id": "PMUL0025-1",
"predicted_belief_state": "<SYS> i haven't found anything. are you interested in a specific cuisine?
<USR> what about a thai restaurant? <bos> restaurant area west <SEP> restaurant food afternoon tea
<SEP> restaurant food thai <eos> ",


"example_id": "PMUL0025-2",
"predicted_belief_state": "<SYS> i have 2. do you prefer the west of town or the centre? <USR>
i would prefer the west please. <bos> restaurant area west <SEP> restaurant food afternoon tea <SEP>
restaurant food thai <SEP> restaurant area west <eos> ",
...
```

A disadvantage of using such a simple turn-level to dialogue-level system can be seen in the second turn of the above example. The system can only append a correction/clarification (in this case `restaurant food thai`) to the dialogue-level belief state rather than replacing the original slot-value pair. This could lead to erroneous suggestions later on in the conversation in some scenarios, particularly if the user changes their mind mid-conversation. What is more, the third turn shows that the clarification statement duplicates the previous slot-value pair on restaurant location. An improvement to this simple DST system could therefore be to be able to overwrite historical slot-value pairs if information from a later turn corrects them.

The scoring of the dialogue-level predicted belief states is again carried out by `$BDIR/multiwoz_dst_score.py`. This time however, the dialogue-level average joint accuracy is measured rather than the turn-level one. This is a more practical metric for real-world use, as in real use-cases the information would be accumulated from the user over multiple turns. The benefit of using a dialogue-level DST system can be seen in Table 3, where both turn-level belief states and dialogue-level belief states are compared against the *dialogue-level* targets.

	Test Set		Development Set
Batch	Turn-level AJA (%)	Dialogue-level AJA (%)	Dialogue-level AJA (%)
80,000	13.41	38.57	41.69
100,000	13.41	34.73	37.75
120,000	13.33	37.19	40.29
140,000	13.34	36.92	41.13
160,000	13.38	39.45	42.26
180,000	13.61	41.26	44.78
200,000	13.60	42.55	46.46

Table 3: Comlumns 2 and 3 represent turn-level and dialogue-level average joint accuracies achieved by the fine-tuned model on the test set when compared to the dialogue-level human annotated targets, after training on variable number of batches. Column 4 then represents dialogue-level average joint accuracy compared to the dialogue-level targets on the development set.

Similar to Table 2 in part 2, the general trend of Table 3 is an increase in both turn and dialogue-level average joint accuracies with number of fine-tuning batches. Also apparent is the large difference between turn-level and dialogue-level accuracies when being scored against the dialogue-level targets. As discussed above, this is to be expected, as the dialogue-level targets tend to contain many more slot-value pairs than can be inferred from a single turn, even for a very good NLU system. Therefore, there is a lot of information absent in the turn-level belief states compared to the dialogue-level ones. This supports the need to move from turn-level belief states of an NLU system to conversation-level belief states of a DST one.

Finally, the drop in dialogue-level average joint accuracy compared to turn-level accuracies of Table 2 in part 2 is due to there now being many more slot-value pairs to compare to the reference, causing greater opportunity for erroneous entries (due to the all-or-nothing joint accuracy scoring procedure and simplicity of the concatenated DST, an error in one turn's output makes the whole DST output erroneous).

5 Exercise GPT2DST.4

5.1 An Adapted DST System:

As discussed in part 3, a drawback of the current DST system is how it does not allow for the correction or change of previous entries in future turns as the historical belief states are uneditable in the simple concatenation regime. This is addressed by adapting `cc-dst.py` to pass in the slots (found in `$GPT2DST/data_preparation/data/multiwoz21/refs/slot_list.json`) to form a dictionary of slot-value pairs for each conversation's dialogue-level belief state. Now if a slot is repeated for a later turn in the conversation, the new value overwrites the previous value rather than just appending it to the end of the linearised belief state string. The effect of this can be seen by revisiting the example output from part 3 when the model had been fine-tuned over 200,000 batches.

```
"example_id": "PMUL0025-0",
"predicted_belief_state": "<USR> can i get a good place for afternoon tea in the westside? <bos>
restaurant area west <SEP> restaurant food afternoon tea <eos>"


"example_id": "PMUL0025-1",
"predicted_belief_state": "<SYS> i haven't found anything. are you interested in a specific cuisine?
<USR> what about a thai restaurant? <bos> restaurant area west <SEP> restaurant food thai <eos>",
```

Notice how in the second turn above the `restaurant food thai` slot-value pair now *overwrites* the previous `restaurant food afternoon tea`, in a way that it did not before in part 3.

With respect to the scoring however, these adaptations do not affect the dialogue-level average joint accuracy, because the scoring procedure automatically only takes the latest value of each slot in the linearised belief state (e.g. `"restaurant food afternoon tea <SEP> restaurant food thai"` from the DST is converted to be analogous to `"restaurant food thai"` only by the scoring script.) This means that as far as the scoring script is concerned, the non-overwriting original DST system effectively overwrites historical slot-value pairs in the same way as the new adapted version. The new DST performance is thus a carbon-copy of Table 3 in part 3, and is given by:

Batch	Test Set Dialogue-level AJA (%)	Dev. Set Dialogue-level AJA (%)
80,000	38.57	41.69
100,000	34.73	37.75
120,000	37.19	40.29
140,000	36.92	41.13
160,000	39.45	42.26
180,000	41.26	44.78
200,000	42.55	46.46

Table 4: Test and development set dialogue-level average joint accuracies of the adapted DST system. These results are exactly the same as with the original concatenation DST system despite the adaptations for the reasons relating to the scoring procedure outlined above.

However, this effect is just a consequence of the way in which scoring is performed, and the adapted DST system would likely still be more useful for practical use due to the fact that it does not carry through erroneous slot-value pairs after a correction is made, removing the risk of system responses being based upon incorrect user requests. There is also the other small advantage that slightly less memory is taken up by the adapted system as each dialogue-level belief state now has a cap on the number of possible slot-value pairs that it can store (equal to the number of slots in the `slot_list.json` file).

5.2 Comparison with UBAR:

A comparison of this DST system can be made with UBAR, a state-of-the-art task-oriented DST system developed using GPT-2 [3]. The reported dialogue-level average joint accuracy by UBAR on the MultiWOZ 2.1 test set was 56.20%, showing a significant improvement over the simple concatenation and adapted DST systems considered in parts 3 and 4 of this report.

A reason for this could be in relation to how UBAR handles the belief states. UBAR diverts from the domain-specific slot-value pair architecture considered in this report by decoupling the domains (such as hotel, restaurant, etc) from the slot names to allow for greater generalisation across similar domains [3]. In addition, a major difference between the DST systems is that UBAR is trained on the conversation-level with greater information used for context in training, such as intermediate belief states and system actions [3]. In comparison, the systems considered in this report are trained on a turn-level, and only the user utterances and system responses are given to the model for fine-tuning.

Indeed, by retraining exactly the same system without this additional context during training and comparing to the previous performance, the authors of [3] were able to show that the intermediate belief states and system actions were more important than solely user and system responses in improving average joint accuracy. Therefore, it would be interesting to incorporate this feature into the DST systems of this report to see how it impacts their performances.

5.3 Model Card:

Model Details: A task-oriented dialogue state tracking system to perform functions such as information retrieval and simple requests such as booking amenities.

Model Training: Natural language understanding performed by a GPT-2 model as distributed by Hugging Face (see `https://huggingface.co/gpt2` for the relevant model card) that has been fine-tuned for 200,000 batches on the MultiWOZ 2.1 data-set [1] to improve performance specific to this task.

Primary Intended Use-cases: To assist users in retrieving information about a range of amenities such as hotels, restaurants, transport, etc, and performing simple actions like booking these amenities. The primary intended users are therefore members of the general public.

Out-of-scope Use-cases: It is not guaranteed that information retrieved by the DST system is factually correct or relevant, and so the use of this system in critical scenarios where correctness of information is an absolute requirement is not supported.

Evaluation and Training Data: Training at the fine-tuning stage is performed on 7,888 dialogues from MultiWOZ 2.1, resulting in 54,791 turns. A held-out development set of 1,000 dialogues with 7,374 turns is used for validation during training. Evaluation is performed on a held-out test set of 999 turns of MultiWOZ 2.1 with 7,368 turns.

The MutliWOZ (Mutli-domain Wizard Of Oz) 2.1 data-set is used due to it being one of the largest existing task-oriented corpora, spanning a wide range of topics. The age of the corpora also means that it has had multiple rounds of data cleaning. The data consists of annotated human-to-human conversations for the specific purpose of training task-oriented DST systems.

For details on the training data used to pre-train the GPT-2 models, see `https://github.com/openai/gpt-2/blob/master/model_card.md`.

Model Performance: The model achieves 42.55% average joint accuracy and 95.91% average slot accuracy on the MultiWOZ 2.1 test set.

Caveats and Biases: Due to the nature of the MultiWOZ data-set being annotated and checked by human annotators, there are likely very few biases and no offensive statements within the MultiWOZ 2.1 data-set. The contents of the training sets used to pre-train the GPT-2 model are known to contain some biases however due to the nature of being scraped from a wide scope of internet sources, and it is possible that these could diffuse into the DST system. For details on GPT-2's biases see `https://github.com/openai/gpt-2/blob/master/model_card.md` and `https://huggingface.co/gpt2`.

6 Conclusion

This report investigated how a modularised dialogue state tracking system can be composed from a GPT-2 based language model that has been fine-tuned on varying numbers of batches of task-specific conversational data to improve performance.

The findings were that around 140,000 batches of fine-tuning using the MultiWOZ 2.1 training set gave the best development loss values, after which overfitting starts to occur and the development loss increases whilst training loss continues to decline. Despite this apparent overfitting, after about 140,000 batches, the turn-level (and dialogue-level) average joint accuracy did continue to improve as the number of batches increased, continuing to increase even up to 200,000 batches.

At this point the test set turn-level average joint accuracy was 72.58% and the dialogue-level 42.55%. It was reasoned in the report that dialogue-level average joint accuracy is a more useful metric for real world use. It is also unsurprising that dialogue-level accuracy is lower than turn-level accuracy (compared with turn-level targets), as for multiple turns within a conversation there are more opportunities for an erroneous slot-value pair than in a single turn.

To move from a turn-level belief state tracker to a dialogue state tracking system a simple concatenation regime was first employed. This involves simply appending the belief state from each turn onto the end of the linearised string that has been generated so far in the conversation. The advantage of this system is its simplicity to implement and reasonable performance. However, the disadvantage is that if an erroneous slot-value pair is generated in one turn and corrected/changed by the user in a later turn, this concatenation regime keeps a history of the incorrect pair for all future turns regardless (i.e. the belief state history is uneditable). As a way around this, a slightly adapted DST was suggested where a slot-pair value can be overwritten in future turns if the slot is detected and a previous entry already exists for it (as it is assumed that the reason for mentioning it again is as a correction or change of mind of the user). It was discussed in the report that this does not affect the average joint accuracy however due to the scoring procedure only taking the most recent entry for each slot in the linearised string anyway. However, it is suggested that in real world use cases, being able to overwrite slot-value pairs within a conversation will have its advantages.

Finally, this DST system was compared to UBAR, a state-of-the-art system proposed in [3]. It was concluded that the distinctly better performance of UBAR compared to our DST was most likely caused by the conversation-level fine-tuning approach that UBAR takes compared to the turn-level one of this DST.

References

[1] Paweł Budzianowski et al. “MultiWOZ–A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling”. In: *arXiv preprint arXiv:1810.00278* (2018).

[2] Alec Radford et al. “Language Models are Unsupervised Multitask Learners”. In: 2019.

[3] Yunyi Yang, Yunhao Li, and Xiaojun Quan. “Ubar: Towards fully end-to-end task-oriented dialog systems with gpt-2”. In: *arXiv preprint arXiv:2012.03539* (2020).