Attention over Parameters
for Dialogue Systems

Andrea Madotto, Zhaojiang Lin, Chien-Sheng Wu, Jamin Shin, Pascale Fung
Centre for Artificial Intelligence Research (CAiRE)
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
[amadotto,zlinao,jmshinaa]@connect.ust.hk
pascale@ece.ust.hk, wu.jason@salesforce.com

Abstract

Dialogue systems require a great deal of different but complementary expertise to assist, inform, and entertain humans. For example, different domains (e.g., restaurant reservation, train ticket booking) of goal-oriented dialogue systems can be viewed as different skills, and so does ordinary chatting abilities of chit-chat dialogue systems. In this paper, we propose to learn a dialogue system that independently parameterizes different dialogue skills, and learns to select and combine each of them through *Attention over Parameters* (*AoP*). The experimental results show that this approach achieves competitive performance on a combined dataset of MultiWOZ [1], In-Car Assistant [2], and Persona-Chat [3]. Finally, we demonstrate that each dialogue skill is effectively learned and can be combined with other skills to produce selective responses.

1 Introduction

Unlike humans who can do both, goal-oriented dialogues [4, 5] and chit-chat conversations [6, 7] are often learned with separate models. A more desirable approach for the users would be to have a single chat interface that can handle both casual talk and tasks such as reservation or scheduling. This can be formulated as a problem of learning different conversational skills across multiple domains. A skill can be either querying a database, generating daily conversational utterances, or interacting with users in a particular task-domain (e.g. booking a restaurant). One challenge of having multiple skills is that existing datasets either focus only on chit-chat or on goal-oriented dialogues. This is due to the fact that traditional goal-oriented systems are modularized [4, 8, 9, 10, 5]; thus, they cannot be jointly trained with end-to-end architecture as in chit-chat. However, recently proposed end-to-end trainable models [11, 12, 13, 14, 15] and datasets [16, 2] allow us to combine goal-oriented [1, 2] and chit-chat [3] into a single benchmark dataset with multiple conversational skills as shown in Table 1.

A straight forward solution would be to have a single model for all the conversational skills, which has shown to be effective to a certain extent by [17] and [18]. Putting aside the performance in the tasks, such fixed shared-parameter framework, without any task-specific designs, would lose controllability and interpretability in the response generation. In this paper, instead, we propose to model multiple conversational skills using the *Mixture of Experts* (MoE) [19] paradigm, i.e., a model that learns and combine independent specialized experts using a gating function.

For instance, each expert could specialize in different dialogues domains (e.g., Hotel, Train, Chit-Chat etc.) and skills (e.g., generate SQL query). A popular implementation of MoE [20, 21] uses a set of linear transformation (i.e., experts) in between two LSTM [22] layers. However, several problems arise with this implementation: 1) the model is computationally expensive as it has to decode multiple times each expert and make the combination at the representation-level; 2) no prior knowledge is

Table 1: An example from the dataset which includes both chit-chat and task-oriented conversations. The model has to predict all the *Sys* turn, which includes SQL query and generating response from a the Memory content, which is dynamically updated with the queries results. The skills are the prior knowledge needed for the response, where Persona refers to chit-chat.

Spk.	Conversation	Skills
Usr:	Can you help me find a cheap 2 star hotel?	
Sys:	`SELECT * FROM hotel WHERE pricerange='cheap' AND stars=2 AND type='hotel'`	SQL+HOTEL
Mem:	(Result table from the SQL query, with **real** entities.)	
Sys:	Arms is a 2 star hotel in the centre. Shall I book a room for you?	HOTEL
Usr:	Yes, thank you!	
Sys:	Sure, for how many people should I book it for?	HOTEL
	. . .	
Sys:	Hello! How are you today?	PERSONA
Usr:	I am good thank you! Do you like to ski? I really like it.	
Sys:	No, I like to code in Haskell. Do you like to code?	PERSONA

injected in the expert selection (e.g., domains); 3) Seq2Seq model has limited ability in extracting information from a Knowledge Base (KB) (i.e., generated by the SQL query) [2], as required in end-to-end task-oriented dialogues systems [16]. The latter can be solved by using more advanced multi-hop models like the Transformer [23], but the remaining two need to be addressed. Hence, in this paper we:

- propose a novel Transformer-based [23] architecture called *Attention over Parameters* (AoP). This model parameterize the conversational skills of end-to-end dialogue systems with independent decoder parameters (experts), and learns how to dynamically select and combine the appropriate decoder parameter sets by leveraging prior knowledge from the data such as domains and skill types;

- proof that AoP is algorithmically more efficient (Appendix A1) compared to forwarding all the Transformer decoders and then mix their output representation, like is normally done in MoE. Figure 1 illustrates the high-level intuition of the difference;

- empirically show the effectiveness of using specialized parameters in a combined dataset of MultiWOZ [1], In-Car Assistant [2], and Persona-Chat [3], which to the best of our knowledge, is the first evaluation of this genre i.e. end-to-end large-scale multi-domains/skills. Moreover, we show that our model is highly interpretable and is able to combine different learned skills to produce compositional responses.

2 Related Work

Dialogue Task-oriented dialogue models [24] can be categorized in two types: module-based [4, 8, 9, 10, 5] and end-to-end. In this paper, we focus on the latter which are systems that train a single model directly on text transcripts of dialogues. These tasks are tackled by selecting a set of predefined utterances [16, 25, 26, 27, 28, 29] or by generating a sequence of tokens [12, 30, 31, 17, 32]. Especially in the latter, copy-augmented models [11, 12, 13, 14, 15] are very effective since extracting entities from a knowledge base is fundamental. On the other hand, end-to-end open domain chit-chat models have been widely studied [6, 7, 33]. Several works improved on the initially reported baselines with various methodologies [34, 15, 35, 36, 37, 38, 39]. Finally, [17] was the first attempt of having an end-to-end system for both task-oriented models and chit-chat. However, the dataset used for the evaluation was small, evaluated only in single domain, and the chit-chat ability was added manually through rules.

Mixture of Expert & Conditional Computation The idea of having specialized parameters, or so-called experts, has been widely studied topics in the last two decades [19, 40]. For instance, different architecture and methodologies have been used such as SVM [41], Gaussian Processes [42, 43, 44],
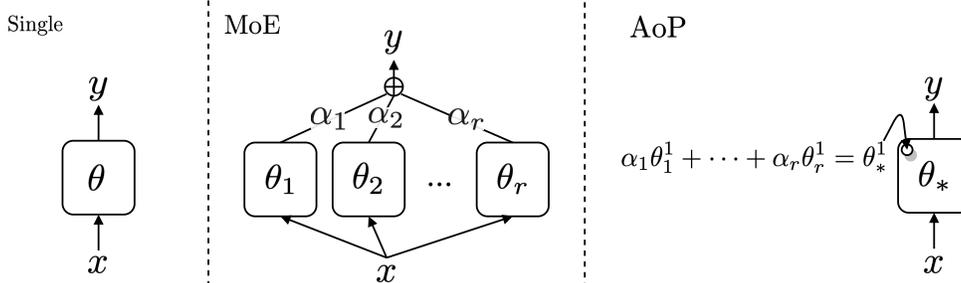
Figure 1: Comparisons between Single model, Mixture of Experts (MoE)[19], and Attention over Parameters (*AoP*).

Dirichlet Processes [45], Hierarchical Experts [46], Infinity Number of Experts [47], and sequential expert addition [48]. More recently, the Mixture Of Expert [20, 21] model was proposed which added a large number of experts between two LSTMs [22]. To the best of our knowledge, none of these previous works applied the results of the gating function to the parameters itself. On the other hand, there are Conditional Computational models which learn to dynamically select their computation graph [49, 50]. Several methods have been used such as reinforcement learning [51], a halting function [52, 53, 54], by pruning [55, 56] and routing/controller function [57]. However, this line of work focuses more on optimizing the inference performance of the model more than specializing parts of it for computing a certain task.

Multi-task Learning   Even though our model processes only input sequence and output sequences of text, it actually jointly learns multiple tasks (e.g. SQL and BOOK query, memory retrieval, and response generation), thus it is also related to multi-task learning [58]. Interested readers may refer to [59, 60] for a general overview on the topic. In Natural Language Processing, multi-task learning has been applied in a wide range of applications such as parsing [61, 62, 63], machine translation in multiple languages [64], and parsing image captioning and machine translation [65]. More interestingly, DecaNLP [18] has a large set of tasks that are cast to question answering (QA), and learned by a single model. In this work, we focus more on conversational data, but in future works, we plan to include these QA tasks.

3   Methodology

We use the standard encoder-decoder architecture and avoid any task-specific designs [12, 13, 14], as we aim to build a generic conversation model for both chit-chat and task-oriented dialogues. More specifically, we use a Transformer [23] for both encoder and decoder.

Let us define the sequence of tokens in the dialogue history as $D = \{d_1, \ldots, d_m\}$ and the dynamic memory content as a sequence of tokens $M = \{m_1, \ldots, m_z\}$. The latter can be the result of a SQL query execution (e.g., table) or plain texts (e.g., persona description), depending on the task. The dialogue history D and the memory M are concatenated to obtain the final input denoted by $X = [D; M] = \{x_1, \ldots, x_{n=m+z}\}$. We then denote $Y = \{y_1, \ldots, y_k\}$ as the sequence of tokens that the model is expected to produce. Without loss of generality, Y can be both plain text and SQL-like queries. Hence, the model has to learn when to issue database queries and when to generate human-like responses. Finally, we define a binary skill vector $V = \{v_1, \ldots, v_r\}$ that specifies the type of skills required to generate Y. This can be considered as a prior vector for learning to select the correct expert during the training[1]. For example, in Table 1 the first response is of type SQL in the Hotel domain, thus the skill vector V will have $v_{\text{SQL}} = 1$ and $v_{\text{Hotel}} = 1$, while all the other skill/domains are set to zero [2]. More importantly, we may set the vector V to have multiple ones to enforce the model to compose skills to achieve a semantic compositionality of different experts.

---

[1]the vector V will be absent during the testing

[2]With the assumption that at each index in V is assigned a semantic skill (e.g. SQL position i)

3

3.1 Encoder-Decoder

To map the input sequence to the output sequence, we use a standard Transformer [23] and denote the encoder and decoder as TRS_{enc} and TRS_{dec}, respectively. The input of a Transformer is the embedded representation of the input words; thus, we define a word embedding matrix $E \in \mathbb{R}^{d \times |V|}$ where d is the embedding size and $|V|$ is the cardinality of the vocabulary. The input X, with its positional embedding (Appendix A2 for more information), are encoded as the following equation:

$$H = \text{TRS}_{enc}(E(X)), \tag{1}$$

where $H \in \mathbb{R}^{d_{model} \times n}$, and E. Then the decoder receives the target sequence shifted by one $Y_{:k-1} = \{\text{<SOS>}, y_1, \ldots, y_k\}$ as the input. Using teacher-forcing [66], the model is trained to produce the correct sequence Y. The output of the decoder is produced as follow:

$$O = \text{TRS}_{dec}(E(Y_{:k-1}), H), \tag{2}$$

where $O \in \mathbb{R}^{d_{model} \times k}$. Finally, a distribution over the vocabulary is generated for each token by an affine transformation $W \in \mathbb{R}^{d_{model} \times |V|}$ followed by a Softmax function.

$$P(Y|X) = \text{Softmax}(O^T W), \tag{3}$$

In addition, $P(Y|X)$ is mixed with the encoder-decoder attention distribution to enable to copy token from the input sequence as in [67]. The model is then trained to minimize a standard cross entropy loss function and at inference time to generate one token at the time in an auto-regressive manner [68]. Hence, the training loss is defined as:

$$\mathcal{L}_{P(Y|X)} = -\sum_{t=1}^{k} \log\left(P(Y|X)_t\left(y_t\right)\right). \tag{4}$$

3.2 Attention over Parameters

The main idea is to produce a single set of parameters for decoder TRS_{dec} by the weighted sum of r independently parameterized decoders. This process is similar to attention [69] where the memories are the parameters and the query is the encoded representation. Let us define $\Theta = [\theta_1, \ldots, \theta_r]$ as the list of parameters for r decoders, since a TRS_{dec} is represented by its parameters θ. Since each θ can be sized in the order of millions, we assign the corresponding key vectors to each θ, similar to key-value memory networks [70]. Thus, we use a key matrix $K \in \mathbb{R}^{d_{model} \times r}$ and a Recurrent Neural Networks (RNN), in this instance a GRU [71], to produce the query vector by processing the encoder output H. The attention weights for each decoders' parameters is computed as follow:

$$q = \text{RNN}(H) \tag{5}$$
$$\alpha = \text{Softmax}(qK) \tag{6}$$

where $q \in \mathbb{R}^{d_{model}}$ and $\alpha \in \mathbb{R}^r$ is the attention vectors where each α_i is the score corresponding to θ_i. Hence, the new set of parameters is computed as follow:

$$\theta^* = \sum_{i}^{r} \alpha_i \theta_i \tag{7}$$

The combined set of parameters θ^* are then used to initialize a new TRS_{dec}, and Equation 2 will be applied to the input based on this. Equation 6 is similar to the gating function proposed in [20, 19], but the resulting scoring vector α is applied directly to the parameter instead of the output representation of each decoder. This indeed makes the computation algorithmically faster since the forward computation is done only once and summing r elements of size $|\theta|$ is linear compared to forwarding r multiplications with the input. Interested readers may refer to Appendix A1 for the proof. Importantly, if we apply α to each of the output representation O^i generated by the TRS_{dec}^i, we end up having a Transformer-based implementation of MoE [3]. We call this model as Attention over Representation (AoR).

---

[3]but with the additional prior over the expert

Moreover, an additional loss term is used to supervise the attention vector α by using the prior knowledge vector V. Since multiple decoder parameters can be selected at the same time, we use a binary cross-entropy to train each α_i. Thus a second loss is defined as:

$$\mathcal{L}_V = -\sum_{i=1}^{r}[v_i \times \log\sigma(qK)_i + (1 - v_i) \times \log(1 - \sigma(qK)_i)] \tag{8}$$

The final loss is the summation of $\mathcal{L}_{P(Y|X)}$ and $\mathcal{L}_V$.

Finally, in AoP, but in general in the MoE framework, stacking multiple layers (e.g., Transformer) leads to models with a large number of parameters, since multiple experts are repeated across layers. An elegant workaround is the Universal Transformer [53], which loops over an *unique* layer and, as shown by [53], holds similar or better performance than a multi-layer Transformer. In our experiment, we report a version of AoP that uses this architecture, which for instance does not add any further parameter to the model.

4 Experiments and Results

4.1 Dataset

To evaluate the performance of our model for different conversational skills, we propose to combine three publicly available datasets: MultiWOZ [1], Stanford Multi-domain Dialogue [2] and Persona-Chat [3].

MultiWOZ (MWOZ) is a human-to-human multi-domain goal-oriented dataset annotated with dialogue acts and states. In this dataset, there are seven domains (i.e., Taxi, Police, Restaurant, Hospital, Hotel, Attraction, Train) and two APIs interfaces: SQL and BOOK. The former is used to retrieve information about a certain domain and the latter is used to book

Figure 2: Datasets statistics

	SMD	MWOZ	Persona
#Dialogues	2425	8,438	12,875
#turns	12,732	115,424	192,690
Avg. turns	5.25	13.68	14.97
Avg. tokens	8.02	13.18	11.96
Vocab	2,842	24,071	20,343

restaurants, hotels, trains, and taxis. We refine this dataset to include SQL/BOOK queries and their outputs using the same annotations schema as [16]. Hence, each response can either be plain text conversation with the user or SQL/BOOK queries, and the memory is dynamically populated with the results from the queries as the generated response is based on such information. This transformation allows us to train end-to-end models that learns how and when to produce SQL queries, to retrieve knowledge from a dynamic memory, and to produce plain text response. A detailed explanation is reported in Appendix A4, together with some samples.

Stanford Multi-domain Dialogue (SMD) is another human-to-human multi-domain goal-oriented dataset that is already designed for end-to-end training. There are three domains in this dataset (i.e., Point-of-Interest, Weather, Calendar). The difference between this dataset and MWOZ is that each dialogue is associated with a set of records relevant to the dialogues. The memory is fixed in this case so the model does not need to issue any API calls. However, retrieving the correct entities from the memory is more challenging as the model has to compare different alternatives among records.

Persona-Chat is a multi-turn conversational dataset, in which two speakers are paired and different persona descriptions (4-5 sentences) are randomly assigned to each of them. For example, "*I am an old man*" and "*I like to play football*" are one of the possible persona descriptions provided to the system. Training models using this dataset results in a more persona consistent and fluent conversation compared to other existing datasets [3]. Currently, this dataset has become one of the standard benchmarks for chit-chat systems, thus, we include it in our evaluation.

For all three datasets, we use the training/validation/test split provided by the authors and we keep all the *real entities* in input instead of using their delexicalized version as in [1, 2]. This makes the task more challenging, but at the same time more interesting since we force the model to produce real entities instead of generic and frequent placeholders. Table 2 summarizes the dataset statistics in terms of number of dialogues, turns, and unique tokens. Finally, we merge the three datasets

Table 2: Results for the goal-oriented responses in both MWOZ and SMD. Last raw, and italicized, are the Oracle results, and bold-faced are best in each setting (w and w/o Universal). Results are averaged among three run (full table in Appendix A6).

Model	F1	BLEU	SQL_{Acc}	SQL_{BLEU}	$BOOK_{Acc}$	$BOOK_{BLEU}$
Seq2Seq	38.37	9.42	49.97	81.75	39.05	79.00
TRS	36.91	9.92	61.96	89.08	46.51	78.41
MoE	38.64	9.47	53.60	85.38	37.23	78.55
AoR	40.36	10.66	69.39	90.64	52.15	81.15
AoP	**42.26**	**11.14**	**71.1**	**90.90**	**56.31**	**84.08**
TRS + U	39.39	9.29	61.80	89.70	50.16	79.05
AoP + U	**44.04**	**11.26**	**74.83**	**91.90**	**56.37**	**84.15**
AoP w/o $\mathcal{L}_V$	38.50	10.50	61.47	88.28	52.61	80.34
AoP+O	46.36	11.99	73.41	93.81	56.18	86.42

obtaining 154,768/19,713/19,528 for training, validation and, test respectively, and a vocabulary size of 37,069 unique tokens.

4.2 Evaluation Metrics

Goal-Oriented For both MWOZ and SMD, we follow the evaluation done by existing works [11, 17, 12, 26]. We use BLEU[4] score [72] to measure the response fluency and Entity F1-Score [30, 17] to evaluates the ability of the model to generate relevant entities from the dynamic memory. Since MWOZ also includes SQL and BOOK queries, we compute the exact match accuracy (i.e., ACC_{SQL} and ACC_{BOOK}) and BLEU score (i.e., $BLEU_{SQL}$ and $BLEU_{BOOK}$). Furthermore, we also report the F1-score for each domain in both MWOZ and SMD.

Chit-Chat We compare perplexity, BLEU score, F1-score [39], and Consistency score of the generate sentences with the human-generated prediction. The Consistency score is computed using a Natural Language Inference (NLI) model trained on dialogue NLI [73], a recently proposed corpus based on Persona dataset. We fine-tune a pre-trained BERT model [74] using the dialogue DNLI corpus and achieve a test set accuracy of 88.43%, which is similar to the best-reported model in [73]. The consistency score is defined as follow:

$$\mathbf{NLI}(u, p_j) = \begin{cases} 1 & \text{if } u \text{ entails } p_j \\ 0 & \text{if } u \text{ is independent of } p_j \\ -1 & \text{if } u \text{ contradicts } p_j \end{cases} \qquad \mathbf{C}(u) = \sum_{j}^{m} \mathbf{NLI}(u, p_j) \qquad (9)$$

where u is a generated utterance and p_j is one sentence in the persona description. In [73], the authors showed that by re-ranking the beam search hypothesis using the DNLI score (i.e., $\mathbf{C}$ score), they achieved a substantial improvement in dialogue consistency. Intuitively, having a higher consistency $\mathbf{C}$ score means having a more persona consistent dialogue response.

4.3 Baselines

In our experiments, we compare Sequence-to-Sequence (*Seq2Seq*) [67], Transformer (*TRS*) [23], Mixture of Expert (*MoE*) [20] and Attention over Representation (*AoR*) with our proposed Attention over Parameters (*AoP*). In all the models, we used the same copy-mechanism as in [67]. In *AoR* instead of mixing the parameters as in Equation 7, we mix the output representation of each transformer decoder (i.e. Equation 2). For all *AoP*, *AoR*, and *MoE*, $r = 13$ is the number of decoders (experts): 2 skills of SQL and BOOK, 10 different domains for MWOZ+SMD, and 1 for Persona-Chat. Furthermore, we include also the following experiments: *AoP* that uses the gold attention vector V, which we refer as *AoP* w/ Oracle (or *AoP* + O); *AoP* trained by removing the $\mathcal{L}_V$ from the optimization (*AoP w/o* $\mathcal{L}_V$); and as aforementioned, the Universal Transformer for both AoP (*AoP* + U) and the standard Transformer (*TRS* + U) (i.e., 6 hops). All detailed model description and the full set of hyper-parameters used in the experiments are reported in Appendix A5.

---

[4]Using the `multi-bleu.perl` script

4.4 Results

Table 2 and Table 3 show the respectively evaluation results in MWOZ+SMD and Persona-Chat datasets. From Table 2, we can identify four patterns. 1) *AoP* and *AoR* perform consistently better then other baselines which shows the effectiveness of combining parameters by using the correct prior V; 2) *AoP* performs consistently, but marginally, better than *AoR*, with the advantage of an algorithmic faster inference; 3) Using Oracle (*AoP*+O) gives the highest performance in all the measures, which shows the performance upper-bound for *AoP*. Hence, the performance gap when not using oracle attention is most likely due to the error in attention α (i.e., 2% error rate). Moreover, Table 2 shows that by removing $\mathcal{L}_V$ (*AoP w/o* $\mathcal{L}_V$) the model performance decreases, which confirms that good inductive bias is important for learning how to select and combine different parameters (experts). Additionally, in Appendix A6, we report the per-domain F1-Score for SQL, BOOK and sentences, and Table 3 and Table 2 with the standard deviation among the three runs.

Furthermore, from Table 3, we can notice that *MoE* has the lowest perplexity and F1-score, but *AoP* has the highest Consistency and BLUE score. Notice that the perplexity reported in [3] is lower since the vocabulary used in their experiments is smaller. In general, the difference in performance among different models is marginal except for the Consistency score; thus, we can conclude that all the models can learn this skill reasonably well. Consistently with the previous results, when $\mathcal{L}_V$ is removed from the optimization, the models' performance decreases.

Figure 3: Results for the Persona-Chat dataset.

Model	Ppl.	F1	C	BLEU
Seq2Seq	39.42	6.33	0.11	2.79
TRS	43.12	7.00	0.07	2.56
MoE	**38.63**	**7.33**	0.19	2.92
AoR	40.18	6.66	0.12	2.69
AoP	39.14	7.00	**0.21**	**3.06**
TRS + U	43.04	**7.33**	0.15	2.66
AoP + U	**37.40**	7.00	**0.29**	**3.22**
AoP w/o $\mathcal{L}_V$	42.81	6.66	0.12	2.85
AoP + O	*40.16*	*7.33*	*0.21*	*2.91*

Finally, in both Table 2 and Table 3, we report the results obtained by using the Universal Transformer, for both AoP and the Transformer. By adding the layer recursion, both models are able to consistently improve all the evaluated measures, in both Persona-Chat and the Task-Oriented tasks. Especially AoP, which achieves better performance than Oracle (i.e. single layer) in the SQL accuracy, and a consistently better performance in the Persona-Chat evaluation.

5 Skill Composition

To demonstrate the effectiveness of our model in learning independent skills and composing them together, we *manually trigger* skills by modifying α and generate 14 different responses for the same input dialogue context. This experiment allows us to verify whether the model accurately captures the meaning of each skill and whether it can properly learn to compose the selected parameters (skills). Table 3 first shows the dialogue history along with the response of *AoP* on the top, and then different responses generated by modifying α (i.e., black cells correspond to 1 in the vector, while the whites are 0). By analyzing Table 3 [5] we can notice that:

- The model learns the correct semantics of each skill. For instance, the *AoP* response is of type SQL and Train, and by deactivating the SQL skill and activating other domain-skills, including Train, we can see that the responses are grammatical and they are coherent with the selected skill semantics. For instance, by just selecting *Train*, the generated answer becomes *"what time would you like to leave?"* which is coherent with the dialogue context since such information has not been yet provided. Interestingly, when *Persona* skill is selected, the generated response is conversational and also coherent with the dialogue, even though it is less fluent.

- The model effectively learns how to compose multiple skills. For instance, when SQL or BOOK are triggered the response produces the correct SQL-syntax (e.g. "SELECT * FROM .." etc.). By also adding the corresponding domain-skill, the model generates the correct query format and attributes relative to the domain type (e.g. in *SQL, Restaurant*, the model queries with the relevant attribute *food* for restaurants).

7

Table 3: Selecting different skills thought the attention vector α results in a skill-consistent response. *AoP* response activates SQL and Train.

Sys: There are lots of trains to choose from! Where are you departing from?
Usr: I am departing from london heading to cambridge.
Sys: What time will you be travelling?
Usr: I need to arrive by 1530.
AoP: SELECT * FROM train WHERE destination="cambridge" AND
day="monday" AND arriveBy < "1530" and departure="london"

SQL	BOOK	Taxi	Restaurant	Hotel	Attraction	Train	Schedule	Persona	*AoP* responses using different parameter combination
		■							I would be happy to book that for you.
			■						I have found you a few to choose from. what time would ..
				■					I have many options for you. is there a certain time you ...
					■				Is there a certain area you would like?
						■			what time would you like to leave?
							■		okay I have two trains what time would you like to cambridge?
								■	Where do you do for work ?
	■	■							BOOK FROM taxi WHERE leaveAt>"1530" AND destination=..
	■		■						BOOK FROM restaurant WHERE time="1530" AND ...
	■			■					BOOK FROM hotel WHERE people="1" AND day="monday"
	■					■			BOOK FROM train WHERE people="1" AND id_booking='
■			■						SELECT * FROM restaurant WHERE food="1530" ...
■				■					SELECT * FROM hotel WHERE type="1530"
■					■				SELECT * FROM attraction WHERE name="departure"

6 Conclusion

In this paper, we propose a novel way to train a single end-to-end dialogue model with multiple composable and interpretable skills. Unlike previous work, that mostly focused on the representation-level mixing [20], our proposed approach, *Attention over Parameters*, learns how to softly combine independent sets of specialized parameters (i.e., making SQL-Query, conversing with consistent persona, etc.) into a single set of parameters. By doing so, we not only achieve compositionality and interpretability but also gain algorithmically faster inference speed. To train and evaluate our model, we organize a multi-domain task-oriented datasets into end-to-end trainable formats and combine it with a conversational dataset (i.e. Persona-Chat). Our model learns to consider each task and domain as a separate skill that can be composed with each other, or used independently, and we verify the effectiveness of the interpretability and compositionality with competitive experimental results and thorough analysis.

Several extensions of this work are possible, for example: incremental learning and zero-shot skill composition. The first, would be similar to [75] where we can add skills through time and learn how to combine it to existing ones. The second, instead, is more related to the semantic compositionality shown in the analysis, where each skill is correctly learned and can be apply to control the generation. An interesting direction would be to learn more general skills (e.g. Machine Translation (MT) or emotional responses), and being able to mix it to existing skills to obtain compositional responses without labeled data.

References

[1] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, 2018.

[2] Mihail Eric, Lakshmi Krishnan, Francois Charette, and Christopher D. Manning. Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 37–49. Association for Computational Linguistics, 2017.

[3] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213. Association for Computational Linguistics, 2018.

[4] Jason D Williams and Steve Young. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422, 2007.

[5] Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[6] Iulian Vlad Serban, Ryan Lowe, Laurent Charlin, and Joelle Pineau. Generative deep neural networks for dialogue: A short review. *arXiv preprint arXiv:1611.06216*, 2016.

[7] Oriol Vinyals and Quoc V Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[8] Chiori Hori, Kiyonori Ohtake, Teruhisa Misu, Hideki Kashioka, and Satoshi Nakamura. Statistical dialog management applied to wfst-based dialog systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.*, pages 4793–4796. IEEE, 2009.

[9] Cheongjae Lee, Sangkeun Jung, Seokhwan Kim, and Gary Geunbae Lee. Example-based dialog modeling for practical multi-domain dialog system. *Speech Communication*, 51(5):466–484, 2009.

[10] Esther Levin, Roberto Pieraccini, and Wieland Eckert. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on speech and audio processing*, 8(1):11–23, 2000.

[11] Mihail Eric and Christopher Manning. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 468–473, Valencia, Spain, April 2017. Association for Computational Linguistics.

[12] Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1468–1478, 2018.

[13] Chien-Sheng Wu, Richard Socher, and Caiming Xiong. Global-to-local memory pointer networks for task-oriented dialogue. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[14] Revanth Reddy, Danish Contractor, Dinesh Raghu, and Sachindra Joshi. Multi-level memory for task oriented dialogs. *arXiv preprint arXiv:1810.10647*, 2018.

[15] Semih Yavuz, Abhinav Rastogi, Guan-lin Chao, Dilek Hakkani-Tür, and Amazon Alexa AI. Deepcopy: Grounded response generation with hierarchical pointer networks. *Conversational AI NIPS workshop*, 2018.

[16] Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. *International Conference on Learning Representations*, abs/1605.07683, 2017.

[17] Tiancheng Zhao, Allen Lu, Kyusong Lee, and Maxine Eskenazi. Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 27–36. Association for Computational Linguistics, August 2017.

[18] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*, 2018.

[19] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, Geoffrey E Hinton, et al. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[20] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[21] Lukasz Kaiser, Aidan N Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *arXiv*, 2017.

[22] Jurgen Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 14 May 1987.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

[24] Jianfeng Gao, Michel Galley, and Lihong Li. Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374. ACM, 2018.

[25] Fei Liu and Julien Perez. Gated end-to-end memory networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1–10, Valencia, Spain, April 2017. Association for Computational Linguistics.

[26] Chien-Sheng Wu, Andrea Madotto, Genta Winata, and Pascale Fung. End-to-end recurrent entity network for entity-value independent goal-oriented dialog learning. In *Dialog System Technology Challenges Workshop, DSTC6*, 2017.

[27] Chien-Sheng Wu, Andrea Madotto, Genta Winata, and Pascale Fung. End-to-end dynamic query memory network for entity-value independent task-oriented dialog. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[28] Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 665–677, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[29] Minjoon Seo, Sewon Min, Ali Farhadi, and Hannaneh Hajishirzi. Query-reduction networks for question answering. *International Conference on Learning Representations*, 2017.

[30] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina Maria Rojas-Barahona, Pei hao Su, Stefan Ultes, David Vandyke, and Steve J. Young. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, 2017.

[31] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, pages 3776–3784, 2016.

[32] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301, 2017.

[33] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*, 2019.

[34] Ilya Kulikov, Alexander H Miller, Kyunghyun Cho, and Jason Weston. Importance of a search strategy in neural dialogue modelling. *arXiv preprint arXiv:1811.00907*, 2018.

[35] Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. Learning from dialogue after deployment: Feed yourself, chatbot! *arXiv preprint arXiv:1901.05415*, 2019.

[36] JM Lucas, F Fernández, J Salazar, J Ferreiros, and R San Segundo. Managing speaker identity and user profiles in a spoken dialogue system. *Procesamiento del lenguaje natural*, 1(43), 2009.

[37] Chaitanya K Joshi, Fei Mi, and Boi Faltings. Personalization in goal-oriented dialog. *arXiv preprint arXiv:1706.07503*, 2017.

[38] Yury Zemlyanskiy and Fei Sha. Aiming to know you better perhaps makes me a more engaging dialogue partner. *CoNLL 2018*, page 551, 2018.

[39] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. The second conversational intelligence challenge (convai2). *arXiv preprint arXiv:1902.00098*, 2019.

[40] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[41] Ronan Collobert, Samy Bengio, and Yoshua Bengio. A parallel mixture of svms for very large scale problems. In *Advances in Neural Information Processing Systems*, pages 633–640, 2002.

[42] Volker Tresp. Mixtures of gaussian processes. In *Advances in neural information processing systems*, pages 654–660, 2001.

[43] Lucas Theis and Matthias Bethge. Generative image modeling using spatial lstms. In *Advances in Neural Information Processing Systems*, pages 1927–1935, 2015.

[44] Marc Deisenroth and Jun Wei Ng. Distributed gaussian processes. In *International Conference on Machine Learning*, pages 1481–1490, 2015.

[45] Babak Shahbaba and Radford Neal. Nonlinear models using dirichlet process mixtures. *Journal of Machine Learning Research*, 10(Aug):1829–1850, 2009.

[46] Bangpeng Yao, Dirk Walther, Diane Beck, and Li Fei-Fei. Hierarchical mixture of classification experts uncovers interactions between brain regions. In *Advances in Neural Information Processing Systems*, pages 2178–2186, 2009.

[47] Carl E Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. In *Advances in neural information processing systems*, pages 881–888, 2002.

[48] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3366–3375, 2017.

[49] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[50] Andrew Davis and Itamar Arel. Low-rank approximations for conditional feedforward computation in deep neural networks. *arXiv*, 2013.

[51] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *ICLR*, 2016.

[52] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

[53] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. *ICLR*, 2019.

[54] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1039–1048, 2017.

[55] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *Advances in Neural Information Processing Systems*, pages 2181–2191, 2017.

[56] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–800, 2018.

[57] Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *International Conference on Learning Representations*, 2018.

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

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

[60] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *Advances in neural information processing systems*, pages 702–710, 2011.

[61] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537, 2011.

[62] Kazuma Hashimoto, Yoshimasa Tsuruoka, Richard Socher, et al. A joint many-task model: Growing a neural network for multiple nlp tasks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1923–1933, 2017.

[63] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Learning what to share between loosely related tasks. *arXiv preprint arXiv:1705.08142*, 2017.

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

[65] Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. In *International Conference on Learning Representations*, 2016.

[66] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[67] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[68] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[69] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[70] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409, 2016.

[71] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[72] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[73] Welleck Sean, Jason Weston, Arthur Szlam, and Kyunghyun Cho. Dialogue natural language inference. *arXiv preprint arXiv:1811.00671*, 2018.

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

[75] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. 2016.

[76] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

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

A Appendix

A.1 Computational Cost AoP

Corollary .1. *The computation cost of Attention over Parameters (AoP) is always lower than Mixture Of Experts (MoE) as long as the processed sequence is longer than 1.*

Proof. Let $f_\theta : \mathbb{R}^d \to \mathbb{R}^n$ a generic function parametrized by θ. Without loss of generality, we define θ as a affine transformation $W \in \mathbb{R}^{d \times n}$. Let $X \in \mathbb{R}^{t \times d}$ a generic input sequence of length t and d dimensional size. Let the set $F = [f_{\theta_1}, \cdots, f_{\theta_r}]$ be the set of r experts. Hence, the operation done by MoE are:

$$\text{MoE}(X) = f_{\theta_1}(X) + \cdots + f_{\theta_r}(X) = XW_1 + \cdots + XW_r \tag{10}$$

Thus the computational cost in term of operation is $\mathcal{O}(rtdn + rtn)$ since the cost of $f_{\theta_i}(X)$ is $\mathcal{O}(tdn)$ and it is repeated r times, and the cost of summing the representation is $\mathcal{O}(rtn)$. On the other hand, the operation done by AoP are:

$$\theta^* = \theta_1 + \cdots + \theta_r = W_1 + \cdots + W_r \tag{11}$$
$$\text{AoP}(X) = f_{\theta^*}(X) = XW^* \tag{12}$$

in this case the computational cost in term of operation is $\mathcal{O}((r + t)dn)$ since the cost of summing the parameters is $\mathcal{O}(rdn)$ and the cost of f_{θ^*} is $\mathcal{O}(tdn)$. Hence, it is easy to verify that if $t > 1$ then:

$$rtdn + rtn \geq (rt)dn \geq (r + t)dn \tag{13}$$

Furthermore, the assumption of using a simple affine transformation W is actually an optimal case. Indeed, assuming that the cost of parameters sum is equal to the number of operation is optimistic, for instance already by using attention the number of operations increases but the number of parameters remains constant. □

A.2 Embedded Representation

Since the model input may include structured data (e.g. DB records) we further define another embedding matrix for encoding the types and the segments as $P \in \mathbb{R}^{d \times |S|}$ where S is the set of positional tokens and $|S|$ its cardinality. P is used to inform the model of the token types such as speaker information (e.g. *Sys* and *Usr*), the data-type for the memory content (e.g. *Miles*, *Traffic* etc.), and segment types like dialogue turn information and database record index [33]. Figure 4 shows an example of the embedded representation of the input. Hence, we denote X_T and X_R as the type and segment tokens for each token in input X, respectively.

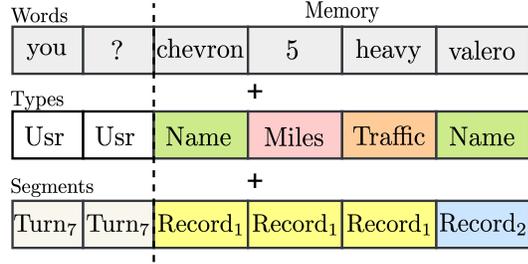

Figure 4: Positional Embedding of the dialogue history and the memory content.

A.3 Attention Visualization

Figure 5 shows the attention vector α over parameters for different generated sentences. In this figure, and by analyzing more examples [5], we can identify two patterns:

- *AoP* learns to focus on the correct skills (i.e., SQL, BOOK) when API-calls are needed. From the first example in Figure 5, we can see that the activations in α are consistent with those in the correct attention vector P. There are also false positives, in which *AoP* puts too high weights on BOOK when the correct response is plain text that should request more information from the user (i.e., *i can help you with that. when would you like to leave the hotel?*). However, we can notice that this example is, in fact, "almost correct" as triggering a booking API call may also be considered a valid response. Meanwhile, the third example also fails to attend to the correct skill, but, in fact, generates a very fluent and relevant response. This is most likely because the answer is simple and generic.

---

[5]Available in supplementary material and later online.

Usr: I would like to stay in an expensive 3 star place with free parking. Is there anything like that?
Ref.: SELECT * FROM hotel WHERE parking='yes' AND pricerange='expensive' AND stars='3'
AoP: SELECT * FROM hotel WHERE parking='yes' AND pricerange='expensive' AND stars='3'

Usr: Thanks. I will also need a taxi from the hotel to the restaurant. Will you handle this?
Ref.: I can help you with that . when would you like to leave the hotel?
AoP: BOOK FROM taxi WHERE destination='finches bed and breakfast' AND departure='ask'

Usr: Thank you for your help!
Ref.: Thank you and enjoy your stay!
AoP: You are welcome. Is there anything else I can help you with?

Figure 5: Attention over Parameters visualization, vector α for different reference (**Ref.**) and *AoP* generated answers. Top rows (**Usr**) are the last utterances from each dialogue contexts.

- The attention often focuses on multiple skills not directly relevant to the task. We observe this pattern especially when there are other skill-related entities mentioned in the context or the response. For example, in the second dialog example in Figure 5, we can notice that *AoP* not only accurately focuses on *taxi* domain, but also has non-negligible activations for *restaurant* and *hotel*. This is because the words "hotel" and "restaurant" are both mentioned in the dialogue context and the model has to produce two entities of the same type (i.e. *finches bed and breakfast* and *ask*).

A.4 Data Pre-Processing

As mentioned in the main article, we convert MultiWOZ into an end-to-end trainable dataset. This requires to add sql-syntax queries when the system includes particular entities. To do so we leverage two annotations such as the state-tracker and the speech acts. The first is used to generate the a well-formed query, including key and attribute, the second instead to decide when to include the query. More details on the dialogue state-tracker slots and slots value, and the different speech acts can be found in [1].

A query is create by the slots, and its values, that has been updated in the latest turn. The SQL query uses the following syntax:

$$\textbf{SELECT} * \textbf{FROM } \textit{domain} \textbf{ WHERE } [\textit{slot\_type} = \textit{slot\_value}]^*$$

Similarly for the booking api BOOK the syntax is the following:

$$\textbf{BOOK FROM } \textit{domain} \textbf{ WHERE } [\textit{slot\_type} = \textit{slot\_value}]^*$$

In both cases the slot values are kept as real entities.

More challenging is to decide when to issue such apis. Speech acts are used to decide by using the "INFORM-DOMAIN" and "RECOMMEND-DOMAIN" tag. Thus any response that include those speech tag will trigger an api if and only if:

- there has been a change in the state-tracker from the previous turn
- the produced query has never been issued before

By a manual checking, this strategy results to be effective. However, as reported by [1] the speech act annotation includes some noise, which is reflected also into our dataset.

The results from the SQL query can be of more that 1K records with multiple attributes. Following [1] we use the following strategy:

- If no speech act INFORM or RECOMMEND and the number of records are more than 5, we use a special token in the memory $< TM >$.
- If no speech act INFORM or RECOMMEND and the number of records are less or equal than 5, we put all the records in memory.

15

- If any speech act INFORM or RECOMMEND, we filter the records to include based on the act value. Notice that this is a fair strategy, since all the resulting record are correct possible answers and the annotators pick-up on of the record randomly [1].

Notice that the answer of a booking call instead, is only one record containing the booking information (e.g. reference number, taxi plate etc.) or "Not Available" token in case the booking cannot made.

A.5 Hyper-parameters and Training

We used a standard Transformer architecture [23] with pre-trained Glove embedding [76]. For the both Seq2Seq and MoE we use Adam [77] optimizer with a learning rate of 1×10^{-3}, where instead for the Transformer we used a warm-up learning rate strategy as in [23]. In both *AoP* and *AoR* we use an additional transformer layer on top the output of the model. Figure 6,7,8 shows the high level design MoE, AoR and AoP respectively. In all the model we used a batch size of 16, and we early stopped the model using the Validation set. All the experiments has been conducted using a single Nvidia 1080ti.

We used a small grid-search for tuning each model. The selected hyper-parameters are reported in Table 4, and we run each experiment 3 times and report the mean and standard deviation of each result.

Model	*d*	d_{model}	*Layers*	*Head*	*Depth*	*Filter*	*GloVe*	*Experts*
Seq2Seq	100	100	1	-	-	-	Yes	-
TRS	300	300	1	2	40	50	Yes	-
MoE	100	100	2	-	-	-	Yes	13
AoP,AoR	300	300	1	2	40	50	Yes	13
TRS/AoP+U	300	300	6	2	40	50	Yes	13

Table 4: Hyper-Parameters used for the evaluations.

A.6 MWOZ and SMD with Std.

Model	F1	BLEU	SQL$_{Acc}$	SQL$_{BLEU}$	BOOK$_{Acc}$	BOOK$_{BLEU}$
Seq2Seq	38.37 ± 1.69	9.42 ± 0.38	49.97 ± 3.49	81.75 ± 2.54	39.05 ± 9.52	79.00 ± 3.63
TRS	36.91 ± 1.24	9.92 ± 0.43	61.96 ± 3.95	89.08 ± 1.23	46.51 ± 5.46	78.41 ± 2.03
MoE	38.64 ± 1.11	9.47 ± 0.59	53.60 ± 4.62	85.38 ± 2.68	37.23 ± 3.89	78.55 ± 2.62
AoR	40.36 ± 1.39	10.66 ± 0.34	69.39 ± 1.05	90.64 ± 0.83	52.15 ± 2.22	81.15 ± 0.32
AoP	$\mathbf{42.26} \pm 2.39$	$\mathbf{11.14} \pm 0.39$	$\mathbf{71.1} \pm 0.47$	$\mathbf{90.90} \pm 0.81$	$\mathbf{56.31} \pm 0.46$	$\mathbf{84.08} \pm 0.99$
TRS + U	39.39 ± 1.23	9.29 ± 0.71	61.80 ± 4.82	89.70 ± 1.40	50.16 ± 1.18	79.05 ± 1.42
AoP + U	$\mathbf{44.04} \pm 0.92$	$\mathbf{11.26} \pm 0.07$	$\mathbf{74.83} \pm 0.79$	$\mathbf{91.90} \pm 1.03$	$\mathbf{56.37} \pm 0.92$	$\mathbf{84.15} \pm 0.32$
AoP w/o $\mathcal{L}_V$	38.50 ± 1.15	10.50 ± 0.55	61.47 ± 0.15	88.28 ± 0.50	52.61 ± 0.56	80.34 ± 0.21
AoP+O	46.36 ± 0.92	11.99 ± 0.03	73.41 ± 0.59	93.81 ± 0.16	56.18 ± 1.55	86.42 ± 0.92

A.7 Persona Result with Std

Model	Ppl.	F1	C	BLEU
Seq2Seq	39.42 ± 1.54	6.33 ± 0.58	0.11 ± 0.06	2.80 ± 0.09
TRS	43.12 ± 1.46	7.00 ± 0.00	0.07 ± 0.16	2.56 ± 0.07
MoE	$\mathbf{38.63} \pm 0.20$	$\mathbf{7.33} \pm 0.05$	0.19 ± 0.16	2.92 ± 0.48
AoR	40.18 ± 0.74	6.66 ± 0.05	0.12 ± 0.14	2.69 ± 0.34
AoP	39.14 ± 0.48	7.00 ± 0.00	$\mathbf{0.21} \pm 0.05$	$\mathbf{3.06} \pm 0.08$
TRS + U	43.04 ± 1.78	$\mathbf{7.33} \pm 0.57$	0.15 ± 0.02	2.66 ± 0.43
AoP + U	$\mathbf{37.40} \pm 0.08$	7.00 ± 0.00	$\mathbf{0.29} \pm 0.07$	$\mathbf{3.22} \pm 0.04$
AoP w/o $\mathcal{L}_V$	42.81 ± 0.01	6.66 ± 0.57	0.12 ± 0.04	2.85 ± 0.21
AoP + O	40.16 ± 0.56	7.33 ± 0.58	0.21 ± 0.14	2.98 ± 0.05

A.8 Domain F1-Score

Sentence	Seq2Seq	MoE	TRS	AoR	AoP	Aop+O
Taxi	71.77	75.97	73.92	76.07	**76.58**	78.30
Police	49.73	49.95	50.24	51.95	**56.61**	52.05
Restaurant	50.20	49.59	48.34	**50.58**	50.47	50.90
Hotel	**46.82**	45.37	43.38	45.51	46.40	44.47
Attraction	37.87	35.21	33.10	36.97	**38.79**	37.51
Train	46.02	41.72	41.28	44.33	**46.32**	45.93
Weather	40.38	27.06	18.97	44.77	**51.94**	55.23
Schedule	35.98	43.94	38.95	32.90	**54.18**	52.99
Navigate	18.57	**21.34**	6.96	12.69	12.18	16.56
BOOK						
Taxi	23.16	32.28	30.70	41.93	**46.66**	43.15
Restaurant	45.02	28.26	49.72	55.70	**58.51**	57.70
Hotel	49.22	31.48	41.61	51.46	**56.62**	57.41
Train	55.86	56.38	57.51	57.51	**59.15**	60.80
SQL						
Police	81.33	0.00	90.66	76.00	**93.33**	100.0
Restaurant	71.58	68.00	75.90	**81.27**	80.43	84.15
Hospital	62.22	15.55	58.89	71.11	**76.67**	83.33
Hotel	45.25	42.09	48.61	56.69	**59.75**	63.75
Attraction	65.48	67.69	65.91	70.61	**76.22**	74.93
Train	30.02	41.01	55.67	66.61	**67.34**	69.50

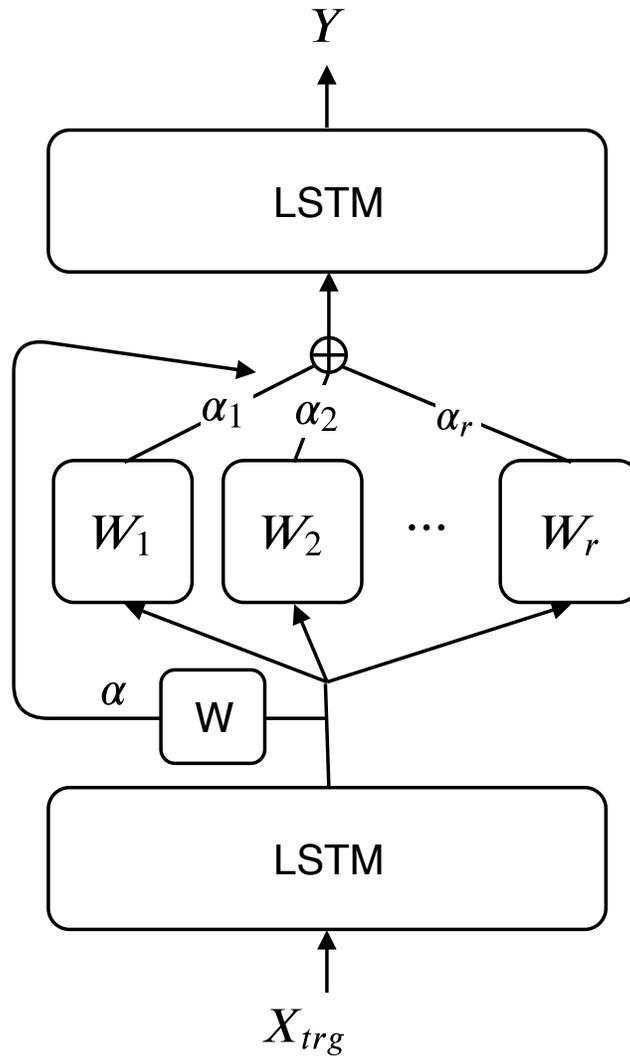Table 5: Per Domain F1 Score.

Figure 6: Mixture of Experts (MoE) [20] model consist of r feed-forward neural network (experts) which are embedded between two LSTM layers, a trainable gating network to select experts.
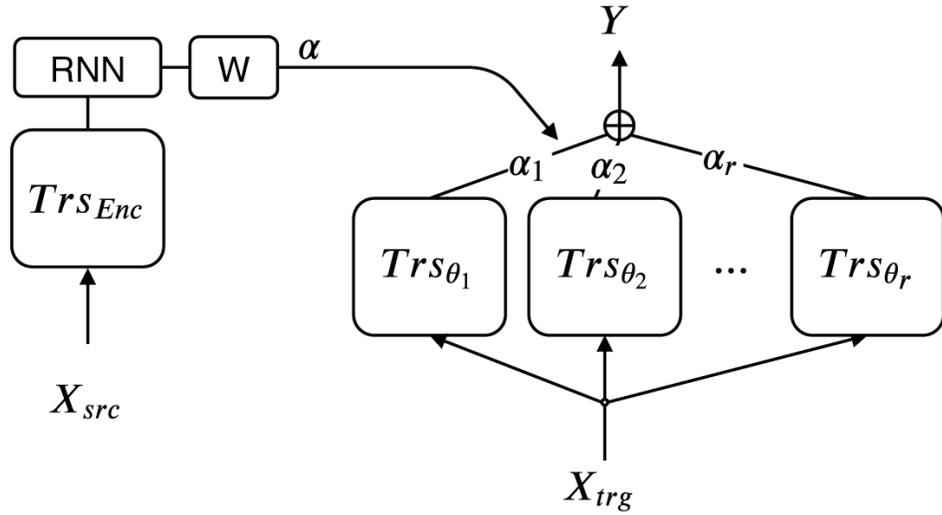
AoR

Figure 7: Attention over Representation (AoR) consist of a transformer encoder which encode the source input and compute the attention over the skills. Then r transformer decoder layers computes r specialized representation and the output response is generated based on the weighted sum the representation. In the figure, we omitted the output layer.
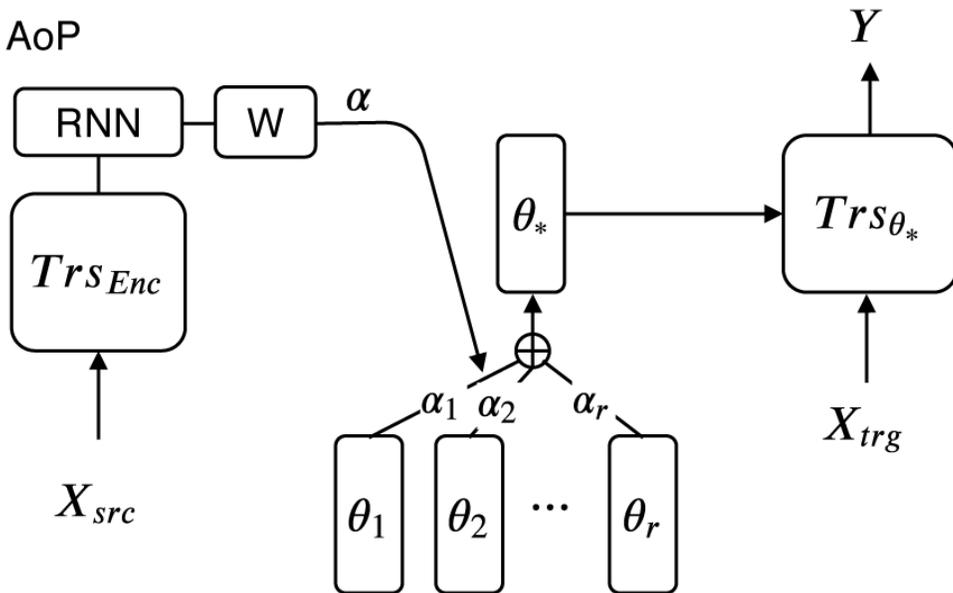
AoP

Figure 8: Attention over Parameters (AoP) consist of a transformer encoder which encode the source input and compute the attention over the skills. Then, r specialized transformer decoder layers and a dummy transformer decoder layer parameterized by the weighted sum of the r specialized transformer decoder layers parameters. In the figure, we omitted the output layer.