UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Retrospective and Prospective Mixture-of-Generators for Task-oriented Dialogue Response Generation

Pei, J.; Ren, P.; Monz, C.; de Rijke, M.

Publication date
2019

Document Version
Submitted manuscript

Link to publication

Citation for published version (APA):
Pei, J., Ren, P., Monz, C., & de Rijke, M. (2019). *Retrospective and Prospective Mixture-of-Generators for Task-oriented Dialogue Response Generation*. (v1 ed.) ArXiv. https://arxiv.org/abs/1911.08151

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:08 Oct 2026

arXiv:1911.08151v1 [cs.CL] 19 Nov 2019

Retrospective and Prospective Mixture-of-Generators for Task-oriented Dialogue Response Generation

Jiahuan Pei and **Pengjie Ren** and **Christof Monz** and **Maarten de Rijke**[1]

Abstract. Dialogue response generation (DRG) is a critical component of task-oriented dialogue systems (TDSs). Its purpose is to generate proper natural language responses given some context, e.g., historical utterances, system states, etc. State-of-the-art work focuses on how to better tackle DRG in an end-to-end way. Typically, such studies assume that each token is drawn from a single distribution over the output vocabulary, which may not always be optimal. Responses vary greatly with different intents, e.g., domains, system actions. We propose a novel mixture-of-generators network (MoGNet) for DRG, where we assume that each token of a response is drawn from a mixture of distributions. MoGNet consists of a chair generator and several expert generators. Each expert is specialized for DRG w.r.t. a particular intent. The chair coordinates multiple experts and combines the output they have generated to produce more appropriate responses.

We propose two strategies to help the chair make better decisions, namely, a retrospective mixture-of-generators (RMoG) and a prospective mixture-of-generators (PMoG). The former only considers the historical expert-generated responses until the current time step while the latter also considers possible expert-generated responses in the future by encouraging exploration. In order to differentiate experts, we also devise a global-and-local (GL) learning scheme that forces each expert to be specialized towards a particular intent using a local loss and trains the chair and all experts to coordinate using a global loss.

We carry out extensive experiments on the MultiWOZ benchmark dataset. MoGNet significantly outperforms state-of-the-art methods in terms of both automatic and human evaluations, demonstrating its effectiveness for DRG.

1 INTRODUCTION

Task-oriented dialogue systems (TDSs) have sparked considerable interest due to their broad applicability, e.g., for booking flight tickets or scheduling meetings [31, 33]. Existing TDS methods can be divided into two broad categories: pipeline multiple-module models [2, 5, 33] and end-to-end single-module models [11, 29]. The former decomposes the TDS task into sequentially dependent modules that are addressed by separate models while the latter proposes to use an end-to-end model to solve the entire task. In both categories, there are many factors to consider in order to achieve good performance, such as user intent understanding [30], dialogue state tracking [36], and dialogue response generation (DRG). Given a *dialogue context* (dialogue history, states, retrieved results from a knowledge base, etc.), the purpose of DRG is to generate a proper natural language response that leads to task-completion, i.e., successfully achieving specific goals, and that is fluent, i.e., generating natural and fluent utterances.

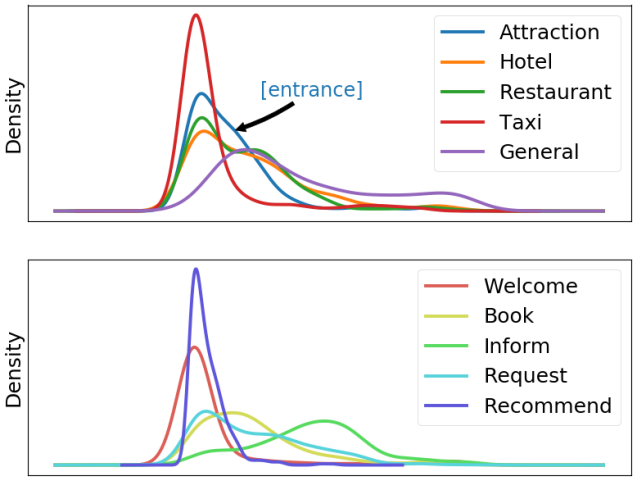

Figure 1: Density of the relative token frequency distribution for different intents (*domains* in the top plot, *system actions* in the bottom plot). We use kernel density estimation[2] to estimate the probability density function of a random variable from a relative token frequency distribution.

Recently proposed DRG methods have achieved promising results [see, e.g., LaRLAttnGRU, 35]. However, when generating a response, all current models assume that each token is drawn from a single distribution over the output vocabulary. This is unreasonable because responses vary greatly with different intents, where intent may refer to domain, system action, or other criteria for partioning responses. To support this claim, consider the training set of the Multi-Domain Wizard-of-Oz (MultiWOZ) benchmark dataset [4], where 67.4% of the dialogues span across multiple domains and all of the dialogues span across multiple types of system actions. We plot the density of the relative token frequency distributions in responses of different intents over the output vocabulary in Fig. 1. Although there is some overlap among different distributions, there are also clear differences. For example, when generating the token $[entrance]$, it has a high probability of being drawn from the distributions for the intent of *booking an attraction*, but not from *booking a taxi*. Thus, we hypothesize that a response should be drawn from a mixture of distributions for multiple intents rather than from a single distribution for a general intent.

[1] University of Amsterdam, The Netherlands, email: {j.pei, p.ren, c.monz, derijke}@uva.nl

[2] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.plot.kde.html

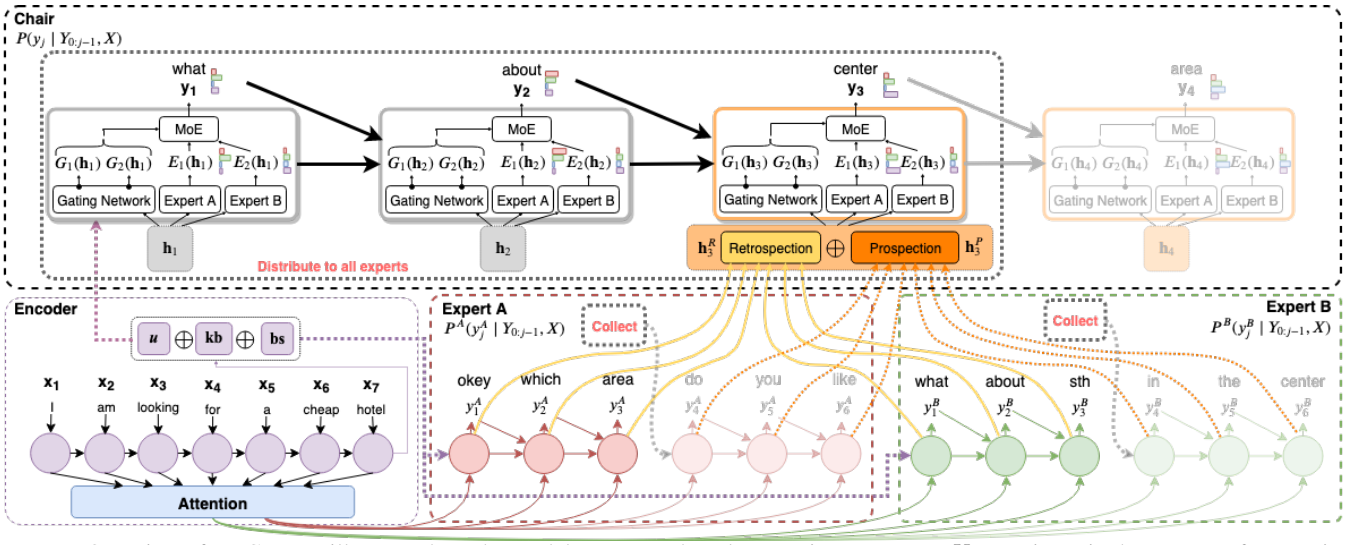

Figure 2: Overview of MoGNet. It illustrates how the model generates the token y_3 given sequence X as an input in the process of generating the whole sequence Y as a dialogue response.

We propose a *mixture-of-generators network* (MoGNet) for DRG, which consists of a *chair* generator and several *expert* generators. Each expert is specialized for a particular intent, e.g., one domain, or one type of action of a system, etc. The chair coordinates multiple experts and generates the final response by taking the utterances generated by the experts into consideration. Compared with previous methods, the advantages of MoGNet are at least two-fold: First, the specialization of different experts and the use of a chair for combining the outputs breaks the bottleneck of a single model [10, 19]. Second, it is more easily *traceable*: we can analyze who is responsible when the model makes a mistake and generates an inappropriate response.

We propose two strategies to help the chair make good decisions, i.e., *retrospective mixture-of-generators* (RMoG) and *prospective mixture-of-generators* (PMoG). RMoG only considers the retrospective utterances generated by the experts, i.e., the utterances generated by all the experts prior to the current time step. However, a chair without a long-range vision is likely to make sub-optimal decisions. Consider, for example, these two responses: "what **day** will you be traveling?" and "what **day** and **time** would you like to travel?" If we only consider these responses until the 2nd token (which RMoG does), then the chair might choose the first response due to the absence of a more long-range view of the important token "time" located after the 2nd token. Hence, we also propose a PMoG, which enables the chair to make full use of the prospective predictions of experts as well.

To effectively train MoGNet, we devise a *global-and-local* (GL) learning scheme. The local loss is defined on a segment of data with a certain intent, which forces each expert to specialize. The global loss is defined on all data, which forces the chair and all experts to coordinate with each other. The global loss can also improve data utilization by enabling the backpropagation error of each data sample to influence all experts as well as the chair.

To verify the effectiveness of MoGNet, we carry out experiments on the MultiWOZ benchmark dataset. MoGNet significantly outperforms state-of-the-art DRG methods, improving over the best performing model on this dataset by 5.64% in terms of overall performance (0.5**Inform*+0.5**Success*+*BLEU*) and 0.97% in terms of response generation quality (*Perplexity*).

The main contributions of this paper are:

- a novel MoGNet model that is the first framework that devises chair and expert generators for DRG, to the best of our knowledge;
- two novel coordination mechanisms, i.e., RMoG and PMoG, to help the chair make better decisions; and
- a GL learning scheme to differentiate experts and fuse data efficiently.

2 MIXTURE-OF-GENERATORS NETWORK

We focus on task-oriented DRG (a.k.a. the context-to-text generation task [4]). Formally, given a current dialogue context $X = (U, B, D)$, where U is a combination of previous utterances, B are the belief states, and D are the retrieved database results based on B, the goal of task-oriented DRG is to generate a fluent natural language response $Y = (y_1, \ldots, y_n)$ that contains appropriate system actions to help users accomplish their task goals, e.g., booking a flight ticket. We propose MoGNet to model the generation probability $P(Y \mid X)$.

2.1 Overview

The MoGNet framework consists of two types of role:

- k **expert generators**, each of which is specialized for a particular *intent*, e.g., a domain, a type of action of a system, etc. Let $\mathcal{D} = \{(X_p, Y_p)\}_{p=1}^{|\mathcal{D}|}$ denote a dataset with $|\mathcal{D}|$ independent samples of (X, Y). Expert-related intents partition $\mathcal{D}$ into k pieces $\mathcal{S} = \{\mathcal{S}_l\}_{l=1}^{k}$, where $\mathcal{S}_l \triangleq \{(X_p^l, Y_p^l)\}_{p=1}^{|\mathcal{S}_l|}$. Then $\mathcal{S}_l$ is used to train each expert by predicting $P^l(Y^l \mid X^l)$. We expect the l-th expert to perform better than the others on $\mathcal{S}_l$.
- a **chair generator**, which learns to coordinate a group of experts to make an optimal decision. The chair is trained to predict $P(Y \mid X)$, where (X, Y) is a sample from $\mathcal{D}$.

Fig. 2 shows our implementation of MoGNet; it consists of three types of components, i.e., a shared context encoder, k expert decoders, and a chair decoder.

2.2 Shared context encoder

The role of the shared context encoder is to read the dialogue context X and construct a representation. We follow Budzianowski et al. [3] and model the current dialogue context as a combination of the user utterance U, belief states B, and retrieved results from the database D.

First, we employ a Recurrent Neural Network (RNN) [7] to map a sequence of input tokens $U = \{w_0, w_1, \ldots, w_m\}$ to hidden vectors $\mathbf{H}^U = \{\mathbf{h}_1^U, \ldots, \mathbf{h}_m^U\}$. The hidden vector $\mathbf{h}_i$ at the i-th step can be represented as:

$$\mathbf{h}_i^U, \mathbf{s}_i = \text{RNN}(\mathbf{w}_i, \mathbf{h}_{i-1}^U, \mathbf{s}_{i-1}), \tag{1}$$

where $\mathbf{w}_i$ is the embedding of the token w_i. The initial state $\mathbf{s}_0$ of the RNN is set to 0.

Then, we represent the current dialogue context $\mathbf{x}$ as a combination of the user utterance representation $\mathbf{h}_m^U$, the belief state vector $\mathbf{h}^B$, and the database vector $\mathbf{h}^D$:

$$\mathbf{x} = \tanh(\mathbf{W}_u \mathbf{h}_m^U + \mathbf{W}_b \mathbf{h}^B + \mathbf{W}_d \mathbf{h}^D), \tag{2}$$

where $\mathbf{h}_m^U$ is the final hidden state from Eq. 1; $\mathbf{h}^B$ is a 0-1 vector with each dimension representing a state (slot-value pair); $\mathbf{h}^D$ is also a 0-1 vector, which is built by querying the database with the current state B. Each dimension of $\mathbf{h}^D$ represents a particular result from the database (e.g., whether a flight ticket is available).

2.3 Expert decoder

Given the current dialogue context X and the current decoded tokens $Y_{0:j-1}$, the l-th expert outputs the probability $P^l(y_j^l \mid Y_{0:j-1}, X)$ over the vocabulary $\mathcal{V}$ at the j-th step by:

$$\begin{aligned} P^l(y_j^l \mid Y_{0:j-1}, X) &= \text{softmax}(\mathbf{U}^T \mathbf{o}_j^l + \mathbf{b}) \\ \mathbf{o}_j^l, \mathbf{s}_j^l &= \text{RNN}(\mathbf{y}_{j-1} \oplus \mathbf{c}_j^l, \mathbf{o}_{j-1}^l, \mathbf{s}_{j-1}^l), \end{aligned} \tag{3}$$

where $\mathbf{U}$ is the parameter matrix and $\mathbf{b}$ is bias; $\mathbf{s}_j^l$ is the state vector, which is initialized by the dialogue context vector from the shared context encoder, i.e., $\mathbf{s}_0^l = \mathbf{x}$; $\mathbf{y}_{j-1}$ is the embedding of the generated token at time step $j-1$; $\oplus$ is the concatenation operation; $\mathbf{c}_j^l$ is the context vector which is calculated with a concatenation attention mechanism [1, 18] over the hidden representations from a shared context encoder as follows:

$$\begin{aligned} \mathbf{c}_j^l &= \sum_{i=1}^{m} \alpha_{ji}^l \mathbf{h}_i \\ \alpha_{ji}^l &= \frac{\exp(w_{ji}^l)}{\sum_{i=1}^{m} \exp(w_{ji}^l)} \\ w_{ji}^l &= \mathbf{v}_l^T \tanh\left(\mathbf{W}_l^T (\mathbf{h}_i \oplus \mathbf{s}_{j-1}^l) + \mathbf{b}_l\right), \end{aligned} \tag{4}$$

where α is a set of attention weights; $\oplus$ is the concatenation operation. $\mathbf{W}_l$, $\mathbf{b}_l$, $\mathbf{v}_l$ are learnable parameters, which are not shared by different experts in our experiments.

2.4 Chair decoder

Given the current dialogue context X and the current decoded tokens $Y_{0:j-1}$, the chair decoder estimates the final token prediction distribution $P(y_j \mid Y_{0:j-1}, X)$ by combining the prediction probabilities from k experts. Here, we consider two strategies to leverage the prediction probabilities from experts, i.e., RMoG and PMoG. The former only considers expert generator outputs from *history* (until $(j-1)$-th time step), which follows the typical neural Mixture-of-Experts (MoE) architecture [24, 26]. We propose the latter to make the chair generator envision the *future* (i.e., after the $(j-1)$-th time step) by exploring expert generator outputs from t extra steps ($t \in [1, n-j], t \in \mathbb{N}$).

Specifically, the chair determines the prediction $P(y_j \mid Y_{0:j-1}, X)$ as follows:

$$\begin{aligned} P(y_j \mid Y_{0:j-1}, X) = \beta_j^C \cdot P(y_j^c \mid Y_{0:j-1}, X) \\ + \sum_{l=1}^{k} (\beta_j^{l,R} + \beta_j^{l,P}) \cdot P(y_j^l \mid Y_{:j-1}^l, X), \end{aligned} \tag{5}$$

where $P(y_j^c \mid Y_{0:j-1}, X)$ is the prediction probability from the chair itself; $P(y_j^l \mid Y_{0:j-1}, X)$ is the prediction probability from expert l; $\beta_j = [\beta_j^C, \beta_j^{l,R}, \beta_j^{l,P}]$ are normalized coordination coefficients, which are calculated as:

$$\begin{aligned} \beta_j &= \frac{\exp(\mathbf{v}^T \mathbf{h}_j)}{\sum_{l=1}^{k} \exp(\mathbf{v}^T \mathbf{h}_l)} \\ \mathbf{h}_j &= \text{MLP}([P(y_j^c \mid Y_{0:j-1}, X), \mathbf{h}_j^R, \mathbf{h}_j^P]). \end{aligned} \tag{6}$$

β_j^C, $\beta_j^{l,R}$ and $\beta_j^{l,P}$ are estimated w.r.t. $P(y_j^c \mid Y_{0:j-1}, X)$, $\mathbf{h}_j^R$ and $\mathbf{h}_j^P$, respectively. $\mathbf{h}_j^R$ is a list of retrospective decoding outputs from all experts, which is defined as follows:

$$\begin{aligned} \mathbf{h}_j^R = P(y_{1:j-1}^1 \mid y_0, X) \oplus \cdots \oplus P(y_{1:j-1}^l \mid y_0, X) \\ \oplus P(y_{1:j-1}^k \mid y_0, X), \end{aligned} \tag{7}$$

where y_0 is a special token "[BOS]" indicating the start of decoding; $P(y_{1:j-1}^l \mid y_0, X)$ is the output of expert l from the 1-st to the $(j-1)$-th step using Eq. 3; $\mathbf{h}_j^P$ is a list of prospective decoding outputs from all experts, which is defined as follows:

$$\begin{aligned} \mathbf{h}_j^P = P(y_{j:j+t}^1 \mid Y_{0:j-1}, X) \oplus \cdots \\ \oplus P(y_{j:j+t}^l \mid Y_{0:j-1}, X) \\ \oplus P(y_{j:j+t}^k \mid Y_{0:j-1}, X), \end{aligned} \tag{8}$$

where $P(y_{j:j+t}^l \mid Y_{0:j-1}, X)$ are the outputs of expert l from the j-th to $(j+t)$-th step. We obtain $P(y_{j:j+t}^l \mid X)$ by forcing expert l to generate t steps using Eq. 3 based on the current generated tokens $Y_{0:j-1}$.

2.5 Learning scheme

We devise a global-and-local learning scheme to train MoGNet. Each expert l is optimized by a localized expert loss defined on $\mathcal{S}_l$, which forces each expert to specialize on one of the portions of data $\mathcal{S}_l$. We use cross-entropy loss for each expert and the joint loss for all experts is as follows:

$$\mathcal{L}_{experts} = \sum_{l=1}^{k} \sum_{(X_p^l, Y_p^l) \in \mathcal{S}_l} \sum_{j=1}^{n} \mu_l y_j^l \log P(y_j^l \mid Y_{0:j-1}^l, X), \tag{9}$$

where $P(y_j^l \mid Y_{0:j-1}^l, X)$ is the token prediction by expert l (Eq. 3) computed on the r-th data sample; y_j^l is a one-hot vector indicating the ground truth token at j.

We also design a global chair loss to differentiate the losses incurred from different experts. The chair can attribute the source of errors to the expert in charge. For each data sample in $\mathcal{D}$, we calculate the combined taken prediction $P(y_j \mid Y_{0:j-1}, X)$ (Eq. 5). Then the global loss becomes:

$$\mathcal{L}_{chair} = \sum_{r=1}^{|\mathcal{D}|} \sum_{j=1}^{n} y_j \log P(y_j \mid Y_{0:j-1}, X). \tag{10}$$

Our overall optimization follows the joint learning paradigm that is defined as a weighted combination of constituent losses:

$$\mathcal{L} = \lambda \cdot \mathcal{L}_{experts} + (1 - \lambda) \cdot \mathcal{L}_{chair}, \tag{11}$$

where λ is a hyper-parameter to regulate the importance between the experts and the chair for optimizing the loss.

3 EXPERIMENTAL SETUP

3.1 Research questions

We seek to answer the following research questions:

(**RQ1**) Does MoGNet outperform state-of-the-art end-to-end single-module DRG models?

(**RQ2**) How does the choice of a particular coordination mechanism (i.e., RMoG, PMoG, or neither of the two) affect the performance of MoGNet?

(**RQ3**) How does the GL learning scheme compare to using the general global learning as a learning scheme?

3.2 Dataset

Our experiments are conducted on the Multi-Domain Wizard-of-Oz (MultiWOZ) [4] dataset. This is the latest large-scale human-to-human TDS dataset with rich semantic labels, e.g., domains and dialogue actions, and benchmark results of response generation.[3] MultiWOZ consists of $\sim$10k natural conversations between a tourist and a clerk. It has 6 specific action-related domains, i.e., *Attraction*, *Hotel*, *Restaurant*, *Taxi*, *Train*, and *Booking*, and 1 universal domain, i.e., *General*. 67.4% of the dialogues are cross-domain which covers 2–5 domains on average. The average number of turns per dialogue is 13.68; a turn contains 13.18 tokens on average. The dataset is randomly split into into 8,438/1,000/1,000 dialogues for training, validation, and testing, respectively.

3.3 Model variants and baseline

We consider a number of variants of the proposed mixture-of-generators model:

- **MoGNet**: the proposed model with RMoG and PMoG and GL learning scheme.
- **MoGNet-P**: the model without prospection ability by removing PMoG coordination mechanism from MoGNet.
- **MoGNet-P-R**: the model removing the two coordination mechanisms and remaining GL learning scheme.
- **MoGNet-GL**: the model that removes GL learning scheme from MoGNet.

See Table 1 for a summary. Without further indications, the *intents* used are based on identifying eight different domains: Attraction, Booking, Hotel, Restaurant, Taxi, Train, General, and UNK.

Table 1: Model variants.

	$\boldsymbol{\beta_j^C}$	$\boldsymbol{\beta_j^{l,R}}$	$\boldsymbol{\beta_j^{l,P}}$	$\boldsymbol{\lambda}$
MoGNet	True	True	True	0.5
MoGNet-P	True	True	False	0.5
MoGNet-P-R	True	False	False	0.5
MoGNet-GL	True	True	True	0.0

β_j^C, $\beta_j^{l,R}$, $\beta_j^{l,P}$ are from Eq. 5. "True" means we preserve it and learn it as it is. "False" means we remove it (set it to 0). λ is from Eq. 11; i our analysis we report two settings, 0.0 and 0.5. See Section 5.2.

To answer RQ1, we compare MoGNet with the following methods that have reported results on this task according to the official leaderboard.[4]

- **S2SAttnLSTM**. We follow the dominant Sequence-to-Sequence (Seq2Seq) model under an encoder-decoder architecture [5] and reproduce the benchmark baseline, i.e., single-module model named S2SAttnLSTM [3, 4], based on the source code provided by the authors.[4]
- **S2SAttnGRU**. A variant of S2SAttnLSTM, with Gated Recurrent Units (GRUs) instead of LSTMs and other settings kept the same.
- **Structured Fusion**. It learns the traditional dialogue modules and then incorporates these pre-trained sequentially dependent modules into end-to-end dialogue models by structured fusion networks [20].
- **LaRLAttnGRU**. The state-of-the-art model [35], which uses reinforcement learning and models system actions as latent variables. LaRLAttnGRU uses ground truth system action annotations and user goals to estimate the rewards for reinforcement learning during training.

3.4 Evaluation metrics

We use the following commonly used evaluation metrics [4, 35]:

- *Inform*: the fraction of responses that provide a correct entity out of all responses.
- *Success*: the fraction of responses that answer all the requested attributes out of all responses.
- *BLEU*: for comparing the overlap between a generated response to one or more reference responses.
- *Score*: defined as $Score = (0.5 * Inform + 0.5 * Success + BLEU) * 100$. This measures the overall performance in term of both task completion and response fluency [20].
- *PPL*: denotes the perplexity of the generated responses, which is defined as the exponentiation of the entropy. This measures how well a probability DRG model predicts a token in a response generation process.

We use the toolkit released by Budzianowski et al. [3] to calculate the metrics.[5] Following their settings, we also use *Score* as the selection criterion to choose the best model on the validation set and report the performance of the model on the test set. We use a paired t-test to measure statistical significance ($p < 0.01$) of relative improvements.

3.5 Implementation details

Theoretically, the training time complexity of each data sample is $\mathcal{O}(n * (k+1) * n)$, where n is the number of response tokens. To reduce the computation cost, we assign $j + t = n$ and compute the expert prediction with Eq. 3. This means that the chair will make a final decision only after all the experts have decoded their final tokens. Thus, the time complexity decreases to $\mathcal{O}(n * (k+1) + n)$.

For a fair comparison, the vocabulary size is the same as Budzianowski et al. [4], which has 400 tokens. Out-of-vocabulary words are replaced with "[UNK]". We set the word embedding size to 50 and all GRU hidden state sizes to 150. We use Adam [13] as our optimization algorithm with hyperparameters $\alpha = 0.005$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. We also apply gradient clipping [22] with range [–5, 5] during training. We use $l2$ regularization to alleviate overfitting, the weight of which is set to 10^{-5}. We set the mini-batch size to 64. We use greedy search to generate the responses during testing. Please note that if a data has multiple intents, then we assign it to each corresponding experts, respectively. The code will be available online after the paper has been published. [6]

4 RESULTS

4.1 Automatic evaluation

We evaluate the overall performance of MoGNet and the comparable baselines on the metrics defined in §3.4. The results are shown in Table 2. First of all, MoGNet outperforms all baselines by a large margin in terms of overall performance metric, i.e., satisfac-

[3] http://dialogue.mi.eng.cam.ac.uk/index.php/corpus/

[4] The Context-to-Text Generation task at https://github.com/budzianowski/multiwoz.

[5] https://github.com/budzianowski/multiwoz.

[6] https://github.com/Jiahuan-Pei/multiwoz-mdrg

tion *Score*. It significantly outperforms the state-of-the-art baseline

Table 2: Comparison results of MoGNet and the baselines.

	BLEU	**Inform**	**Success**	**Score**	**PPL**
S2SAttnLSTM	18.90%	71.33%	60.96%	85.05	**3.98**
S2SAttnGRU	18.21%	81.50%	68.80%	93.36	4.12
Structured Fusion [20]	16.34%	82.70%	72.10%	93.74	–
LaRLAttnGRU [35]	12.80%	82.78%	**79.20%**	93.79	5.22
MoGNet	**20.13%***	**85.30%***	73.30%	**99.43***	4.25

Bold face indicates leading results. Significant improvements over the best baseline are marked with * (paired t-test, $p < 0.01$).

LaRLAttnGRU by 5.64% (*Score*) and 0.97 (*PPL*). Thus, MoGNet not only improves the satisfaction of responses but also improves the quality of the language modeling process. MoGNet also achieves more than 6.70% overall improvement over the benchmark baseline S2SAttnLSTM and its variant S2SAttnGRU. This proves the effectiveness of the proposed MoGNet model.

Second, LaRLAttnGRU achieves the highest performance in terms of *Success*, followed by MoGNet. However, it results in a 7.33% decrease in *BLEU* and a 2.56% decrease in *Inform* compared to MoGNet. Hence, LaRLAttnGRU is good at answering all requested attributes but not as good at providing more appropriate entities with high fluency as MoGNet. LaRLAttnGRU tends to generate more slot values to increase the probability of answering the requested attributes. Take an extreme case as an example: if we force a model to generate all tokens with slot values, then it will achieve an extremely high *Success* but a low *BLEU*.

Third, S2SAttnLSTM is the worst model in terms of overall performance (*Score*). But it achieves the best *PPL*. It tends to generate frequent tokens from the vocabulary which exhibits better language modeling characteristics. However, it fails to provide useful information (the requested attributes) to meet the user goals. By contrast, MoGNet improves the user satisfaction (i.e., *Score*) greatly and achieves response fluency by taking specialized generations from all experts into account.

4.2 Human evaluation

To further understand the results in Table 2, we conducted a human evaluation of the generated responses from S2SAttnGRU, LaRLAttnGRU, and MoGNet. We ask workers on Amazon Mechanical Turk (AMT)[7] to read the dialogue context, and choose the responses that satisfy the following criteria: (i) *Informativeness* measures whether the response provides appropriate information that is requested by the user query. No extra inappropriate information is provided. (ii) *Consistency* measures whether the generated response is semantically aligned with the ground truth response. (iii) *Satisfactory* measures whether the response has a overall satisfactory performance promising both *Informativeness* and *Consistency*. As with existing studies [20], we sample one hundred context-response pairs to do human evaluation. Each sample is labeled by three workers. The workers are asked to choose either all responses that satisfy the specific criteria or the "NONE" option, which denotes none of the responses satisfy the criteria. To make sure that the annotations are of high quality, we calculate the fraction of the responses that satisfy each criterion out of all responses that passes the *golden test*. That is, we only consider the data from the workers who have chosen the golden response as an answer.

The results are displayed in Table 3. MoGNet performs better than S2SAttnGRU and LaRLAttnGRU on *Informativeness* because it frequently outputs responses that provide richer information (compared with S2SAttnGRU) and fewer extra inappropriate information (compared with LaRLAttnGRU). MoGNet obtains the best results, which means MoGNet is able to generate responses that are semantically similar to the golden responses with large overlaps. The results of LaRLAttnGRU outperforms S2SAttnGRU in all cases except for *Satisfactory* under the strict condition ($\geqslant 2$). This reveals that balancing between *Informativeness* and *Consistency* makes it difficult for the mturk workers to assess the overall quality measured by *Satisfactory*. In this case, MoGNet receives the most votes on *Satisfactory* under the strict condition ($\geqslant 2$) as well as the loose condition ($\geqslant 1$). This shows that the workers consider the responses from MoGNet more appropriate than the other two models with a high degree of agreement. To sum up, MoGNet is able to generate user-favored responses in addition to the improvements for automatic metrics.

Table 3: Results of human evaluation.

	S2SAttnGRU		LaRLAttnGRU		MoGNet	
	$\geqslant 1$	$\geqslant 2$	$\geqslant 1$	$\geqslant 2$	$\geqslant 1$	$\geqslant 2$
Informativeness	56.79%	31.03%	76.54%	44.83%	**80.25%**	**53.45%**
Consistency	45.21%	23.53%	71.23%	39.22%	**80.82%**	**50.98%**
Satisfactory	26.79%	25.00%	44.64%	21.88%	**60.71%**	**37.50%**

Bold face indicates the best results. $\geqslant n$ means that at least n AMT workers regard it as a good response w.r.t. *Informativeness*, *Consistency* and *Satisfactory*.

4.3 Coordination mechanisms

In Table 4 we contrast the effectiveness of different coordination mechanisms. We can see that MoGNet-P loses 4.32% overall performance with a 0.62% decrease of *BLEU*, 5.90% decrease of *Inform* and 1.50% decrease of *Success*. This shows that the prospection design of the PMoG mechanism is beneficial to both task completion and response fluency. Especially, most improvements come from providing more correct entities while improving generation fluency. MoGNet-P-R reduces 2.62% *Score* with 1.97% lower of *BLEU*, 0.2% lower of *Inform* and 1.10% of *Success*. Thus, the MoGNet framework is effective thanks to its design with two types of roles: the chair and the experts.

Table 4: The impact of coordination mechanisms.

	BLEU	**Inform**	**Success**	**Score**	**PPL**
MoGNet	20.13%	85.30%	73.30%	99.43	4.25
MoGNet-P	19.51%	79.40%	71.80%	95.11	4.19
MoGNet-P-R	<u>18.16%</u>	85.10%	72.20%	96.81	4.12

<u>Underlined results</u> indicate the worst results with a statistically significant decrease compared to MoGNet (paired t-test, $p < 0.01$).

4.4 Learning scheme

We use MoGNet-GL to refer to the model that removes the GL learning scheme from MoGNet and uses the general global learning instead. MoGNet-GL results in a sharp reduction of 6.95% overall performance with 0.80% of *BLEU*, 6.90% of *Inform* and 5.40% of *Success*. The main improvement is attributed to the strong task completion ability. This shows the effectiveness and importance of the GL learning scheme as it encourages each expert to specialize on a particular intent while the chair prompts all experts to coordinate with each other.

5 ANALYSIS

In this section, we explore MoGNet in more detail. In particular, we examine (i) whether the intent partition affects the performance of

[7] https://www.mturk.com/

Table 5: Impact of the learning scheme.

	BLEU	Inform	Success	Score	PPL
MoGNet	20.13%	85.30%	73.30%	99.43	4.25
MoGNet-GL	19.33%	78.40%	67.90%	92.48	3.97

Underlined results indicate the worst results with a statistically significant decrease compared with MoGNet (paired t-test, $p < 0.01$).

MoGNet (§5.1); (ii) whether the improvements of MoGNet could simply be attributed to having a larger number of parameters (§5.2); (iii) how the hyper-parameter λ (Eq. 11) affects the performance of MoGNet (§5.2); and (iv) how RMoG, PMoG and GL influence DRG using a small case study (§5.3).

5.1 Intent partition analysis

As we mentioned above, the responses vary a lot for different intents which are differentiated by the domain and the type of system action. Therefore, we experiment with two types of intents as shown in Table 6.

Table 6: Two groups of intents that are divided by domains and the type of system actions.

Type	Intents
Domain	Attraction, Booking, Hotel, Restaurant, Taxi, Train, General, UNK.
Action	Book, Inform, NoBook, NoOffer, OfferBook, OfferBooked, Select, Recommend, Request, Bye, Greet, Reqmore, Welcome, UNK.

To address (i), we compared two ways of partitioning intents. MoGNet-domain and MoGNet-action denote the intent partitions w.r.t. domains and system actions, respectively. MoGNet-domain has 8 intents (domains) and MoGNet-action has 14 intents (actions), as shown in Table 6. The results are shown in Table 7. MoGNet consistently outperforms the baseline S2SAttnGRU for both ways of partitioning intents. Interestingly, MoGNet-domain greatly outperforms MoGNet-action. We believe there are two reasons: First, the system actions are not suitable for grouping intents because some partition subsets are hard to be distinguished from each other, e.g., *OfferBook* and *OfferBooked*. Second, some system actions only have a few data samples, simply not enough to specialize the experts. The results show that different ways of partitioning intents may greatly affect the performance of MoGNet. Therefore, more effective intent partition methods, e.g., adaptive implicit intent partitions, need to be explored in future work.

Table 7: Results of MoGNet with two intent partition ways.

	BLEU	Inform	Success	Score	PPL
MoGNet-domain	**20.13%**	**85.30%**	**73.30%**	**99.43**	**4.25**
MoGNet-action	17.28%	79.40%	69.70%	91.83	4.48

Bold face indicates the best results.

5.2 Hyper-parameter analysis

To address (ii), we show the results of MoGNet and S2SAttnGRU with different hidden sizes in Fig. 3. S2SAttnGRU outperforms MoGNet when the number of parameters is less than 0.6e7. However, MoGNet achieves much better results with more parameters. Most importantly, the results from both models show that a larger number of parameters does not always mean better performance, which shows that the improvement of MoGNet is not simply due to more parameters.

To address (iii), we report the *Score* values of MoGNet with different values of λ (Eq. 11), as shown in Fig. 4. When $\lambda = 0$, no expert is trained on a particular intent. When $\lambda = 1$, the model ignores the global loss, i.e., the RMoG and PMoG mechanisms are not used and the chair is only trained as a general expert. We can see that these two settings decrease the performance greatly which further verifies the effectiveness of RMoG and PMoG as well as the MoGNet framework. We also note that the performance of MoGNet is quite stable when $\lambda \in [0.1, 0.7]$ with the best performance for $\lambda = 0.7$. Hence, MoGNet is not very sensitive to the hyper-parameter λ.

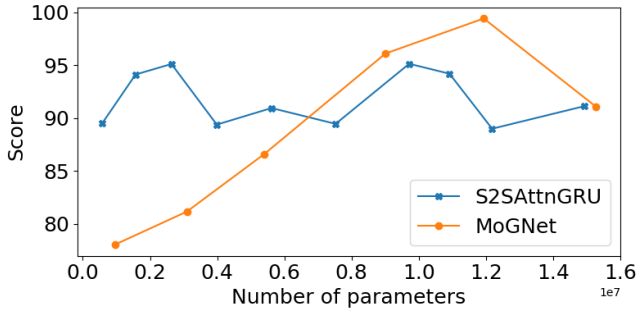

Figure 3: *Score* of MoGNet and S2SAttnGRU with different number of parameters.

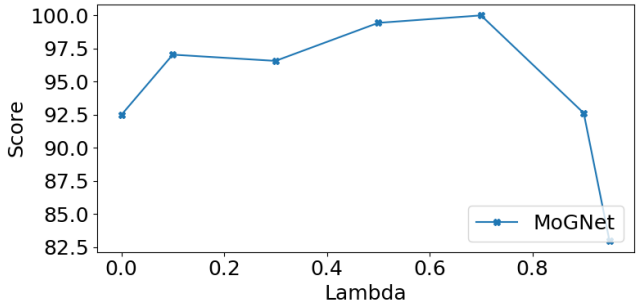

Figure 4: *Score* of MoGNet with different values of hyper-parameter λ.

5.3 Case study

To address (iv), we select an example to illustrate the influence of RMoG, PMoG, and GL. Table 8 exhibits the responses generated by comparable baselines (i.e., S2SAttnGRU, LaRLAttnGRU) and MoGNet variants as in Table 4. In red we highlight the tokens that show the differences in terms of task completion. Generally, MoGNet can generate more appropriate and meaningful responses. Specifically, without PMoG, MoGNet-P and MoGNet-P-R ignore the fact that the attribute *time* is important for searching a train ticket (1st turn) and omit the exact departure time ([value_time]) of the train (2nd turn). Without GL, MoGNet-GL ignores the primary time information need *day* (1st turn) and omits the implicit need of [value_price] (4th turn). There are also some low-quality cases, e.g., MoGNet and the baselines occasionally generate redundant and lengthy responses, because none of them has addressed this issue explicitly during training.

6 RELATED WORK

Traditional models for DRG [8, 32] decompose the task into sequentially dependent modules, e.g., Dialogue State Tracking (DST) [36], Policy Learning (PL) [34], and Natural Language Generation (NLG) [21]. Such models allow for targeted failure analyses, but inevitably incur upstream propagation problems [5]. Recent work views DRG as a source-to-target transduction problem, which maps

Table 8: Example responses from MoGNet with the ablation settings in a 4-turn dialogue.

Turn	User	System
1	i am looking for a train departing from [value_place] that will arrive at [value_place].	**S2SAttnGRU**: what day would you like to travel ? **LaRLAttnGRU**: there are [value_count] trains leaving [value_place] on [value_day] . what day would you like to travel ?
		MoGNet: what day and time would you like to travel? **MoGNet-P**: what day will you be traveling? **MoGNet-P-R**: what day will you be traveling? **MoGNet-GL**: there are [value_count] trains ... do you have a time you would like to depart?
2	i would like to leave on [value_day] and arrive no later than [value_time].	**S2SAttnGRU**: [train_id] will arrive at [value_time] . **LaRLAttnGRU**: i have train [train_id] leaving at [value_time] and arriving at [value_time] . would you like me to book this for you ?
		MoGNet: [train_id] leaves at [value_time] and arrives by [value_time] . **MoGNet-P**: [train_id] will arrive at [value_time]. **MoGNet-P-R**: the [train_id] arrives at [value_time]. **MoGNet-GL**: i have train [train_id] that leaves at [value_time] and arrives at [value_time]. +*All*: would you like me to book that for you?
3	... can i please have the reference number?	*All*: ... how many tickets do you need?
4	could you book [value_count] tickets for me and send me the reference number?	**S2SAttnGRU**: i have booked you [value_count] tickets on the [train_id] . your reference number is [train_reference]. is there anything else i can help you with ? **LaRLAttnGRU**: i have booked [value_count] tickets on [train_id] . the reference number is [train_reference] .
		MoGNet: i have booked [value_count] tickets for you. the total fee is [value_price] gbp payable at the station. your reference number is [train_reference]. is there anything else i can help you with ? **MoGNet-P**: i have booked [value_count] tickets. the total fee is [value_price] gbp payable at the station. your reference number is [train_reference]. **MoGNet-P-R**: booking was successful , the total fee is [value_price] gbp payable at the station. reference number is [train_reference]. **MoGNet-GL**: i have booked [value_count] tickets for you. the reference number is [train_reference]. is there anything else i can help you with ?

a *dialogue context* to a *response* [11, 17, 30]. Sordoni et al. [27] show that using an RNN to generate text conditioned on dialogue history results in more natural conversations. Later improvements include the addition of attention mechanisms [16, 28], modeling the hierarchical structure of dialogues [25], or jointly learning belief spans [15]. Strengths of these methods include global optimization and easier adaptation to new domains [5].

The studies listed above all assume that each token of a response is sampled from a single distribution given complex dialogue context. In contrast, MoGNet uses multiple cooperating modules (expert and chair generators), which exploits the specialization capabilities of different experts and the generalization capability of a chair. Work most closely related to ours in terms of modeling multiple experts includes [6, 12, 14, 23]. Le et al. [14] integrate a chat model with a question answering model using an LSTM-based mixture-of-experts method. Their model is similar to MoGNet-GL-P (without PMoG and GL) except that they simply use two implicit expert generators that are not specialized on particular intents. Guo et al. [12] introduce a mixture-of-experts to use the data relationship between multiple domains for binary classification and sequence tagging. Sequence tagging aims to generate a set of fixed labels; DRG generates diverse appropriate response sequence. The differences between MoGNet and these two approaches are three-fold: First, MoGNet consists of a group of modules including a chair generator and several expert generators; this design addresses the module interdependence problem since each module is independent from the others. Second, the chair generator alleviates the error propagation problem because it is able to manage the overall errors through an effective learning scheme. Third, the models of those two approaches cannot be directly applied to task-oriented DRG. The recently published HDSA [6] slightly outperforms MoGNet on *Score* (+0.07), but it overly relies on BERT [9] and graph structured dialog acts. MoGNet follow the same modular TDS framework [23], but it preforms substantially better due to fitting the expert generators with both retrospection and prospection abilities and adopting the GL learning scheme to conduct more effective learning.

7 CONCLUSION AND FUTURE WORK

In this paper, we propose a novel mixture-of-generators network (MoGNet) model with different coordination mechanisms, namdely, RMoG and PMoG, to enhance dialogue response generation. We also devise a GL learning scheme to effectively learn MoGNet. Experiments on the MultiWOZ benchmark demonstrate that MoGNet significantly outperforms state-of-the-art methods in terms of both automatic and human evaluations. We also conduct analyses that confirm the effectiveness of MoGNet, the RMoG and PMoG mechanisms, as well as the GL learning scheme.

As to future work, we plan to devise more fine-grained expert generators and to experiment on more datasets to test MoGNet. In addition, MoGNet can be advanced in many directions: First, better mechanisms can be proposed to improve the coordination between chair and expert generators. Second, it would be interesting to study how to do intent partition automatically. Third, it is also important to investigate how to avoid redundant and lengthy responses in order to provide a better user experience.

REFERENCES

[1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[2] A. Bordes and J. Weston. Learning end-to-end goal-oriented dialog. In *ICLR*, 2017.

[3] P. Budzianowski, I. Casanueva, B.-H. Tseng, and M. Gasic. Towards end-to-end multi-domain dialogue modelling. Technical report, Cambridge University, 2018.

[4] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gasic. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *EMNLP*, pages 5016–5026, 2018.

[5] H. Chen, X. Liu, D. Yin, and J. Tang. A survey on dialogue systems: Recent advances and new frontiers. *ACM SIGKDD Explorations Newsletter*, 19(2):25–35, 2017.

[6] W. Chen, J. Chen, P. Qin, X. Yan, and W. Y. Wang. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. *arXiv preprint arXiv:1905.12866*, 2019.

[7] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[8] P. Crook, A. Marin, V. Agarwal, K. Aggarwal, T. Anastasakos, R. Bikkula, D. Boies, A. Celikyilmaz, S. Chandramohan, Z. Feizollahi, R. Holenstein, M. Jeong, O. Z. Khan, Y.-B. Kim, E. Krawczyk, X. Liu, D. Panic, V. Radostev, N. Ramesh, J.-P. Robichaud, A. Rochette, S. L., and R. Sarikaya. Task completion platform: A self-serve multi-domain goal oriented dialogue platform. In *NAACL*, pages 47–51, 2016.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, pages 4171–4186, 2019.

[10] T. G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, pages 1–15, 2000.

[11] M. Eric, L. Krishnan, F. Charette, and C. D. Manning. Key-value retrieval networks for task-oriented dialogue. In *SIGDIAL*, pages 37–49, 2017.

[12] J. Guo, D. J. Shah, and R. Barzilay. Multi-source domain adaptation with mixture of experts. In *EMNLP*, pages 4694–4703, 2018.

[13] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[14] P. Le, M. Dymetman, and J.-M. Renders. LSTM-based mixture-of-experts for knowledge-aware dialogues. *arXiv preprint arXiv:1605.01652*, 2016.

[15] W. Lei, X. Jin, M.-Y. Kan, Z. Ren, X. He, and D. Yin. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *ACL*, pages 1437–1447, 2018.

[16] J. Li, M. Galley, C. Brockett, G. P. Spithourakis, J. Gao, and B. Dolan. A persona-based neural conversation model. In *ACL*, pages 994–1003, 2016.

[17] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky. Adversarial learning for neural dialogue generation. In *EMNLP*, pages 2157–2169, 2017.

[18] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, pages 1412–1421, 2015.

[19] S. Masoudnia and R. Ebrahimpour. Mixture of experts: a literature survey. *Artificial Intelligence Review*, 42(2):275–293, 2014.

[20] S. Mehri, T. Srinivasan, and M. Eskenazi. Structured fusion networks for dialog. In *SIGDIAL*, pages 165–177, 2019.

[21] F. Mi, M. Huang, J. Zhang, and B. Faltings. Meta-learning for low-resource natural language generation in task-oriented dialogue systems. In *IJCAI*, pages 3151–3157, 2019.

[22] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *ICML*, pages 1310–1318, 2013.

[23] J. Pei, P. Ren, and M. de Rijke. A modular task-oriented dialogue system using a neural mixture-of-experts. In *SIGIR Workshop on Conversational Interaction Systems*, 2019.

[24] P. Schwab, D. Miladinovic, and W. Karlen. Granger-causal attentive mixtures of experts: Learning important features with neural networks. In *AAAI*, pages 4846–4853, 2019.

[25] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, pages 3776–3784, 2016.

[26] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.

[27] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J.-Y. Nie, J. Gao, and B. Dolan. A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT*, pages 196–205, 2015.

[28] O. Vinyals and Q. Le. A neural conversational model. In *ICML Deep Learning Workshop*, 2015.

[29] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. R. Barahona, P.-H. Su, S. Ultes, and S. Young. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, pages 438–449, 2017.

[30] T.-H. Wen, D. Vandyke, N. Mrkšić, M. Gasic, L. M. R. Barahona, P.-H. Su, S. Ultes, and S. Young. A network-based end-to-end trainable task-oriented dialogue system. In *EACL*, pages 438–449, 2017.

[31] J. D. Williams, K. Asadi, and G. Zweig. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *ACL*, pages 665–677, 2017.

[32] Z. Yan, N. Duan, P. Chen, M. Zhou, J. Zhou, and Z. Li. Building task-oriented dialogue systems for online shopping. In *AAAI*, pages 4618–4626, 2017.

[33] S. Young, M. Gašić, B. Thomson, and J. D. Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

[34] Z. Zhang, M. Huang, Z. Zhao, F. Ji, H. Chen, and X. Zhu. Memory-augmented dialogue management for task-oriented dialogue systems. *TOIS*, 37(3):34, 2019.

[35] T. Zhao, K. Xie, and M. Eskenazi. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *NAACL*, pages 1208–1218, 2019.

[36] V. Zhong, C. Xiong, and R. Socher. Global-locally self-attentive encoder for dialogue state tracking. In *ACL*, pages 1458–1467, 2018.