Joint Generation and Bi-Encoder for Situated Interactive MultiModal Conversations

Xin Huang,[1] **Chor Seng Tan,**[1] **Yan Bin Ng,**[2] **Wei Shi,**[1]
Kheng Hui Yeo,[1] **Ridong Jiang,**[1] **Jung-jae Kim**[1]
{huangx2, tan_chor_seng, shi_wei, yeokh, rjiang, jjkim}@i2r.a-star.edu.sg
ng_yan_bin@ihpc.a-star.edu.sg

Abstract

This paper presents our work on the fourth track in Dialog State Tracking Challenge 9 – Situated Interactive MultiModal Conversations (SIMMC) challenge, which aims at building virtual assistants that can handle multimodal inputs and perform multimodal actions. For this challenge, we propose an end-to-end encoder-decoder model based on BART for generating outputs of action prediction, response generation, and dialogue state tracking tasks in a single string, and another model based on Bi-encoders for response retrieval task. Our models came in the first place for the action prediction and response retrieval tasks, and the second place for the response generation and dialogue state tracking tasks, achieving the first position of overall ranking in the challenge. In particular, our Bi-encoder models for the response retrieval task significantly outperformed the other entries of the challenge's official evaluation. Furthermore, our models show similar performance on the two test datasets (devtest dataset whose ground-truth are released publicly and test-std dataset whose ground-truth are not released publicly), which shows the robustness of our models.

1 Introduction

The Situated Interactive MultiModal Conversations (SIMMC) challenge is one of the tracks in the 9th Dialog State Technology Challenge (DSTC 9), aiming at enabling the conversational AI community to develop virtual assistants that can process situated multimodal context that a user and an assistant co-observe, and provide outputs in multiple modalities (Crook et al. 2019; Moon et al. 2020; Gunasekara et al. 2020). The context is in the form of a co-observed image or a VR environment which gets updated dynamically based on dialog flow and assistant actions, in contrast to existing literature that posit roles of primary and secondary observers (Antol et al. 2015; Das et al. 2017; Kottur et al. 2019; De Vries et al. 2017). The assistant actions span across a broad multimodal action space (e.g. ROTATE, SEARCH, ADD_TO_CART), shifting the focus of the visual dialogue research from the token or phrase-level understanding of visual scenes to the task-level understanding of dialogues, given complex multimodal context. Assistant actions can be enacted on the object-level (e.g. changing the view of the object in the scene) as well as scene level, in contrast to existing dialogue-based image retrieval tasks (Saha, Khapra, and Sankaranarayanan 2018; Guo et al. 2018) which are limited to changes of the visual scene.

The SIMMC challenge has a total of 4 sub-tasks, namely 1) assistant action selection, 2) response generation in natural language, 3) response retrieval from a candidate pool, and 4) dialogue state tracking (DST). We approach the challenge as a joint generation learning task except the response retrieval task, though the first and last sub-tasks (action selection, DST) are classification tasks and use unnatural keywords (e.g. ADD_TO_CART) as labels, inspired by the recent progress of adapting natural language generation (NLG) for various natural language processing (NLP) applications, which had not been considered as generation tasks (Raffel et al. 2020; Brown et al. 2020). In particular, we develop encoder-decoder models, whose encoders analyze the situated multimodal context and whose decoders jointly generate the outputs of the three tasks in a sequence so that the models can learn about the three tasks synergistically. We call the models *joint generation* models.

As for the response retrieval task, we adopt Bi-encoder methods based on pre-trained language model (Humeau et al. 2020), which separately encode an instance of situated multimodal context and each candidate response from a pool and learn to rank the candidates with respect to the situated multimodal context. We call the models trained by the methods *response retrieval* models. The joint generation models and the response retrieval models are trained and evaluated independently.

2 Task Descriptions

The SIMMC datasets are based on shopping experiences between a user and a shopping assistant due to the rich multimodal interactions that occur in such settings. In addition to an interactive dialogue, the shopping assistant also manip-

ulates the environment to display items from the shopping inventory to assist the user. As such, the shopping assistant is required to understand the user's utterance using the dialogue history and the state of the environment (which is provided as a multimodal context), and produce a multimodal response, including updating the co-observed environment to convey meaningful information.

Two datasets with slightly different setups and modalities were provided, set in the fashion and furniture domains respectively. The datasets combined have about 13K dialogues and about 169K utterances, and were collected through the SIMMC platform (Crook et al. 2019). The fashion dataset is grounded in real-world clothing items (e.g. jacket, dress), while the furniture dataset is set in a virtual reality (VR) environment with furniture items which can be manipulated (e.g. rotate, zoom in) by the assistant. Fine-grained annotations are provided for each item to allow for both end-to-end and component-level modeling. The user is, in the beginning of a dialogue, presented either with a high-level directive (e.g. "shop for a table") for the furniture dataset or with a randomly selected "seed" item from the product catalog for the fashion dataset, and browses and asks for recommendations with help from the shopping assistant, which has access to a broader catalog of items (or products). The ground-truth for product recommendations (or appearances) is provided. The user and assistant utterances are also accompanied by dialogue act labels and text spans for the corresponding attributes.

The organizers split the SIMMC dataset into train, dev, devtest, and test-std subsets for both the fashion and furniture domains. The devtest set is the publicly available test set for measuring model performance and results outside of the challenge, while the test-std is used as the main test set for official evaluation of the challenge, with the ground-truth results not publicly disclosed. Performance for the test-std set is evaluated on a per-turn basis instead of across the entire dialogue like the devtest set. We will report the performance of all our models on the devtest set and the performance of our models submitted for the official evaluation on the test-std set in this paper.

Sub-Task #1 – Structural API Call Prediction

The first sub-task involves predicting the assistant action (a_t) as an API call along with the necessary arguments, using dialogue history (H_t), current user utterance (U_t), and the multimodal context (M_t) as inputs. Action prediction is cast as a round-wise, multi-class classification problem over the set of APIs. The evaluation is based on such measures as action accuracy, attribute accuracy, and action perplexity.

Sub-Task #2 – Response Generation

The second sub-task aims to output assistant response utterances given a multi-turn context through response generation, and is evaluated based on the relevance of the assistance response A_t for the current turn. The BLEU-4 score is used to measure the closeness between the generated response and the ground-truth response.

Sub-Task #2 – Response Retrieval

The third sub-task aims to retrieve assistant response given a multi-turn context from a candidate pool of 100 responses, and the recall@1 (R@1), recall@5 (R@5), recall@10 (R@10), mean rank, and mean reciprocal rank (MRR) are used for evaluation.

Sub-Task #3 – Dialogue State Tracking (DST)

The last sub-task is to systematically track the dialogue acts and the associated slot pairs for the conversations across multiple turns. Using current user utterance, dialogue context, and multimodal context as inputs, the model will output the belief state for the current user utterance. Intent and slot F1 accuracies are the evaluation metrics for this sub-task. In addition, annotated labels used for the evaluation include coreferences.

3 Methods

Baseline Models

The challenge organizers provided the following four baseline models: 1) a History-Agnostic Encoder (HAE) that ignores dialogue context H_t and encodes only the user utterance through an LSTM for downstream components, 2) a Hierarchical Recurrent Encoder (HRE) (Serban et al. 2016) that models dialogues at two hierarchical recurrence levels of utterance and turn, 3) a Memory Network (MN) encoder (Sukhbaatar et al. 2015) that treats dialogue history H_t as a collection of memory units and then selectively attends to them with the current utterance u_t, 4) a Transformer-based History-Agnostic Encoder (T-HAE) which is similar to the HAE except with Transformer units (Vaswani et al. 2017) instead of LSTMs. The results of these baseline models were provided for benchmarking purposes.

Joint Generation

We use encoder-decoder models to predict the assistant API action Act and arguments Act_{arg} (Sub-Task #1), system response R (Sub-Task #2 Response Generation), a list of belief state actions $I^{(1)}, ..., I^{(N)}$ and associated belief state slots $I^{(1)}_{arg}, ..., I^{(N)}_{arg}$ (Sub-Task #3), related to current user utterance at each user turn. The input of the model at each user turn is user utterance U_t, dialogue history H_t and multimodal context M_t which includes text description of the visual objects mentioned in the dialogue. The target output

Domain	Method	Sub-task 1			Sub-task 2	Sub-task 3	
		Act. Acc	Att. Acc	Act. Perp	BLEU-4	Slot F1	Intent F1
Fashion	TD-IDF	78.1%	57.9%	3.51	-	-	-
	HAE	81.0%	60.2%	1.75	0.059	-	-
	HRE	81.9%	62.1%	1.76	0.079	-	-
	MN	81.6%	61.6%	**1.74**	0.065	-	-
	T-HAE	81.4%	62.1%	1.78	0.051	-	-
	GPT2	-	-	-	-	60.6%	61.1%
Fashion	MTN	81.2%	71.7%	-	0.098	64.8%	66.5%
	SimpleTOD	82.9%	74.9%	-	0.076	65.7%	61.3%
	Bert2Bert	85.7%	81.6%	-	0.099	68.6%	70.7%
	Bert2Share	85.9%	81.4%	-	0.095	70.8%	72.5%
	Bert2GPT2	85.7%	81.6%	-	0.090	68.6%	70.7%
	Bert2DistilGPT2	83.6%	75.0%	-	0.072	50.7%	64.6%
	BART-base	86.0%	81.5%	-	0.108	73.0%	73.5%
	BART-large	**86.5%**	**81.9%**	2.88	**0.115**	**74.0%**	**75.2%**
Furniture	TD-IDF	77.1%	57.5%	2.59	-	-	-
	HAE	79.7%	53.6%	1.70	0.059	-	-
	HRE	80.0%	54.7%	**1.66**	0.079	-	-
	MN	79.2%	53.3%	1.71	0.065	-	-
	T-HAE	78.4%	53.6%	1.83	0.051	-	-
	GPT2	-	-	-	-	63.9%	69.5%
Furniture	MTN	73.5%	70.0%	-	0.082	72.2%	77.9%
	SimpleTOD	74.0%	63.1%	-	0.058	73.3%	54.6%
	Bert2Bert	78.0%	63.1%	-	0.079	73.8%	79.4%
	Bert2GPT2	78.3%	66.6%	-	0.077	64.3%	80.0%
	Bert2DistilGPT2	79.7%	65.0%	-	0.085	73.1%	81.5%
	BART-base	79.9%	**70.7%**	-	0.099	78.6%	82.9%
	BART-large	**80.5%**	69.4%	4.05	**0.105**	**79.9%**	**83.6%**

Table 1: Evaluation results of joint generation models against the SIMMC devtest dataset in fashion and furniture domains

of the joint generation model is a single string, combining the output strings of the three sub-tasks as follows:

$$Act\ [Act_{arg}]\ [SEP1]\ I^{(1)}\ [I^{(1)}_{arg}], ..., I^{(i)}\ [I^{(i)}_{arg}]\ [SEP2]\ R$$

, where the $[SEP1]$ and $[SEP2]$ are two special tokens in order to separate the outputs of different sub-tasks from each other. We show that the joint generation model performs better than models separately trained for the sub-tasks.

We adapt the following Transformer-based encoder-decoder models for the SIMMC dataset:

- **Bert2Bert** (Rothe, Narayan, and Severyn 2020): The encoder is a pre-trained BERT model which consists of 12 layers and 12 attention heads, where hidden size is 768. The decoder is another pre-trained BERT model of the same size as the encoder, but its weights are not synchronized with those of the encoder and are fine-tuned independently from the encoder. In addition, the decoder includes a cross-attention sub-layer.
- **Bert2Share** (Rothe, Narayan, and Severyn 2020): The encoder and the decoder share the same pre-trained BERT model and its weights, while the decoder additionally has a cross-attention sub-layer.
- **Bert2GPT2** (Rothe, Narayan, and Severyn 2020): The encoder is a pre-trained BERT model. The decoder is a pre-trained GPT2 model which consists of 12 layers, 12 attention heads and a cross-attention sub-layer, where hidden size is 768.
- **Bert2DistilGPT2** (Rothe, Narayan, and Severyn 2020): The encoder is a pre-trained BERT model. The decoder is a pre-trained distilled version of GPT2 model which consists of 6 layers, 12 attention heads and a cross-attention sub-layer, where hidden size is 768.
- **MTN** (Le et al. 2019): The encoder is a text sequence encoder of Multimodal Transformer Network (MTN) model (Le et al. 2019). The decoder is an auto-regressive decoder of the MTN model. They are not pre-trained, but trained from scratch with the SIMMC data. We do not use the video and caption encoders from the MTN model since the SIMMC datasets have only text data.
- **SimpleTOD** (Hosseini-Asl et al. 2020): SimpleTOD uses a single, causal language model jointly trained end-to-end on all generation sub-tasks as a single sequence prediction problem. The underlying pre-trained model used is GPT2.
- **BART** (Lewis et al. 2020): The encoder and the decoder are those of the BART model (Lewis et al. 2020), but the loss of fine-tuning the model for the SIMMC datasets is calculated only based on the errors of the next token prediction task, not involving the reconstruction loss of the denoising step of the original BART model. We use both the pre-trained BART-base and BART-large models for the experiments.

Domain	Model	Sub-task 1			Sub-task 2	Sub-task 3	
	Act. Acc	Att. Acc	Act. Perp	BLEU-4	Slot F1	Intent F1	
Fashion	Baseline	80.7%	67.4%	**1.84**	0.058	61.6%	56.3%
	BART-base	85.4%	**81.6%**	2.91	0.106	73.9%	71.6%
	BART-large	**85.8%**	80.7%	2.90	**0.116**	**76.0%**	**72.9%**
Furniture	Baseline	**77.8%**	60.0%	**1.93**	0.064	63.4%	68.4%
	BART-base	75.3%	**68.6%**	4.11	0.076	78.9%	82.60%
	BART-large	76.8%	67.0%	4.08	**0.100**	**81.5%**	**84.1%**

Table 2: Joint generation models performance on test-std set in fashion and furniture domains

Method	Type	Sub-task 1		Sub-task 2	Sub-task 3	
		Act. Acc	Att. Acc	BLEU-4	Slot F1	Intent F1
MTN	Individual	59.8%	64.4%	0.089	62.6%	62.6%
Bert2Bert	Individual	79.2%	65.3%	-	70.0%	72.1%
BART-base	Individual	83.0%	75.0%	-	71.8%	72.8%
MTN	Joint	81.2%	71.7%	0.098	64.8%	66.5%
Bert2Bert	Joint	85.7%	**81.6%**	0.099	68.6%	70.7%
BART-base	Joint	**86.0%**	81.5%	**0.108**	**73.0%**	**73.5%**

Table 3: Performance comparison between joint learning and individual learning models in fashion domain

Source Text Encoder The sequence of concatenated source text $X^{(i)} = x_1, ..., x_i$ that combines a user utterance $U^{(i)}$ at the i-th turn, a dialogue history $U^{(0)}, ..., U^{(i-1)}$ at previous turns and multimodal contexts $m_{ctx}^{(i)}$ is fed into the *encoder* of a model $(Encoder)$ to get the last layer hidden representation of the input texts, $h_{src}^{(i)}$:

$$X^{(i)} = \left[U^{(0)}, \cdots, U^{(i-1)}, U^{(i)}, m_{ctx}^{(i)}\right] \quad (1)$$

$$h_{src}^{(i)} = Encoder\left(X^{(i)}\right) \quad (2)$$

Target Text Decoder Given the hidden representation output of the encoder $h_{src}^{(i)}$, the goal of the decoder $(Decoder)$ is to generate the next token (y_m) given previous target text sequence $Y_{m-1}^{(i)} = y_1, ..., y_{m-1}$ which includes the information of assistant API action Act, Act_{arg}, system response R, and a list of belief states $I^{(1)}I_{arg}^{(1)}, ..., I^{(N)}I_{arg}^{(N)}$. We assume that the $Decoder$ has an innate multi-head cross-attention layer $CrossAttn$, where the hidden representation of the previous target string $h_{tgt}^{(i)}$ attends to the hidden representation of the encoder's last layer $h_{src}^{(i)}$, as follows (Vaswani et al. 2017):

$$h_{tgt}^{(i)} = Decoder\left(Y_{m-1}^{(i)}\right) \quad (3)$$

$$h'^{(i)}_{tgt} = CrossAttn\left(h_{tgt}^{(i)}, h_{src}^{(i)}\right) \quad (4)$$

The output of the attention layer $h'^{(i)}_{tgt}$ is used to proceed the decoding process. We apply a linear layer for language modeling after the last layer of the decoder to get probabilities for the next token. A cross-entropy loss is used for fine-tuning the model.

Response Retrieval Model

For the Response Retrieval task in Sub-Task #2, we adapt the Bi-encoder and Poly-encoder architectures, which have been shown good results for multi-sentence scoring tasks (Humeau et al. 2020). We used the fine-tuned encoders of the BERT and BART models from the joint learning tasks for the two encoder architectures.

Bi-encoder Bi-encoders are a broad class of models that map the input and candidate responses separately into a common feature space whereby their similarity is measured. Examples of such methods include using memory networks (Zhang et al. 2018), transformers (Dinan et al. 2019), LSTMs (Lowe et al. 2015), and CNNs (Kadlec, Schmid, and Kleindienst 2015) to encode the input and candidate responses. In our case, both the input context at the i^{th} turn $\left(X^{(i)}\right)$ and each candidate response of the turn $\left(m_{cand}^{(i,j)}\right)$ are encoded into vectors $y_{ctxt}^{(i)}$ and $y_{cand}^{(i,j)}$ separately by a pre-trained Transformer encoder:

$$y_{ctxt}^{(i)} = T\left(X^{(i)}\right) \quad (5)$$

$$y_{cand}^{(i,j)} = T\left(m_{cand}^{(i,j)}\right) \quad (6)$$

, where T is a fine-tuned Transformer from the joint learning sub-tasks. The models score the candidate response with regard to the input context by calculating the dot product of the context and response vectors $\left(y_{ctxt}^{(i)} \cdot y_{cand}^{(i,j)}\right)$. The models are trained to minimize the cross-entropy loss.

Poly-encoder While Poly-encoders (Humeau et al. 2020) also represent each candidate response as a single vector like the Bi-encoders, they encode the input context jointly

Domain	Model	Pre-trained model	Parameter	R@1	R@5	R@10	Mean Rank	MRR
Fashion	LSTM			5.3%	11.4%	16.5%	46.9	0.102
	HAE			10.5%	25.3%	34.1%	33.5	0.190
	HRE			16.3%	33.1%	41.7%	27.4	0.253
	MN			16.1%	31.0%	39.4%	29.3	0.245
	T-HAE	GPT-2		10.3%	23.2%	31.1%	37.1	0.178
	Bi-Encoder	BERT (fine-tuned)		46.5%	82.9%	94.1%	3.44	0.621
		BART (pre-trained)		43.8%	82.6%	94.7%	3.41	0.602
		BART (fine-tuned)		**53.3%**	**88.6%**	**96.3%**	**2.81**	**0.681**
	Poly-Encoder	BERT (fine-tuned)	m=16	46.0%	82.7%	94.3%	3.54	0.616
		BART (pre-trained)	m=16	45.7%	84.0%	95.3%	3.27	0.619
		BART (fine-tuned)	m=8	52.7%	88.0%	**96.3%**	2.84	0.676
		BART (fine-tuned)	m=16	52.9%	88.2%	96.2%	2.84	0.678
		BART (fine-tuned)	m=32	51.5%	87.6%	96.0%	2.92	0.667
		BART (fine-tuned)	m=64	52.6%	88.1%	96.2%	2.86	0.676
Furniture	LSTM			4.1%	11.1%	17.3%	46.4	0.094
	HAE			12.9%	28.9%	38.4%	31.0	0.218
	HRE			13.8%	30.5%	40.2%	30.0	0.229
	MN			15.3%	31.8%	42.2%	29.1	0.244
	T-HAE	GPT-2		8.5%	20.3%	28.9%	37.9	0.156
	Bi-Encoder	BERT (fine-tuned)		48.1%	84.2%	93.5%	3.48	0.635
		BART (pre-trained)		41.6%	80.5%	93.2%	3.87	0.582
		BART (fine-tuned)		47.7%	84.1%	94.0%	3.43	0.631
	Poly-Encoder	BERT (fine-tuned)	m=16	45.2%	81.4%	93.2%	3.69	0.610
		BART (pre-trained)	m=16	45.1%	82.6%	93.6%	3.60	0.611
		BART (fine-tuned)	m=8	46.8%	83.6%	93.9%	3.47	0.625
		BART (fine-tuned)	m=16	**48.4%**	**84.3%**	**94.2%**	**3.39**	**0.639**
		BART (fine-tuned)	m=32	45.2%	82.5%	93.5%	3.62	0.613
		BART (fine-tuned)	m=64	43.3%	81.6%	93.0%	3.77	0.598

Table 4: Response Retrieval Task performance on devtest set in fashion and furniture domains

with the candidate in order to allow further joint learning from pair of input context and candidate response. They represent the input context as m vector representations $\left(y_{ctxt}^{(i)(1)}, \cdots, y_{ctxt}^{(i)(m)}\right)$, instead of just one as in the Bi-encoder, and then obtain the context vector $y_{ctxt}^{(i)}$ as the query $y_{cand}^{(i,j)}$ attends to the m context vector representations as values, as follows:

$$y_{ctxt}^{(i)} = \sum_k w_k y_{ctxt}^{(i)(k)} \tag{7}$$

, where

$$(w_1, ..., w_m) = softmax\left(y_{cand}^{(i,j)} \cdot y_{ctxt}^{(i)(1)}, ..., y_{cand}^{(i,j)} \cdot y_{ctxt}^{(i)(m)}\right) \tag{8}$$

Poly-encoders obtain the m context vectors as follows:

$$y_{ctxt}^{(i)(k)} = \sum_l w_l^{c_k} h_l \tag{9}$$

$$(w_1^{c_k}, ..., w_N^{c_k}) = softmax\left(c_k \cdot h_1, ..., c_k \cdot h_N\right) \tag{10}$$

, where $(c_1, ..., c_m)$ are m learnable context codes, and $(h_1, \cdots, h_N)$ are the vectors produced by a pre-trained Transformer encoder over N input context tokens.

The m context codes are randomly initialized and learned during fine-tuning, N is the number of tokens, and m is a hyperparameter $(m < N)$ that controls the tradeoff between the inference speed and performance.

The scoring is also done by $\left(y_{ctxt}^{(i)} \cdot y_{cand_i}^{(i,j)}\right)$ as in the Bi-encoder. This architecture has been shown to improve performance over the Bi-encoder without compromising much on prediction speed (Humeau et al. 2020).

4 Experiment

Joint Generation Tasks

For the joint generation tasks, we chose the hyperparameters of learning rate $\alpha = 5 * 10^{-5}$, number of epochs $E = 10$, and the Adam optimizer with $\epsilon = 10^{-8}$ and $\beta_1 = 0.9, \beta_2 = 0.999$, after a number of experiments. A linear learning rate scheduler is applied to the optimizer with warm up steps as 0.

Table 1 summarizes the evaluation results of the aforementioned encoder-decoder models as well as the baseline models against the SIMMC devtest dataset in the fashion and furniture domains. For Sub-Task #1, the results are evaluated with the accuracy of action and attributes as well as perplexity of action. For Sub-Task #2, the results are evaluated with the BLEU-4 of system response in natural lan-

Domain	Model	R@1	R@5	R@10	Mean Rank	MRR
Fashion	Baseline	4.5%	14.8%	22.1%	42.1	0.113
	Bi-Encoder BART	**51.3%**	**87.8%**	**96.3%**	**3.17**	**0.667**
	Poly-Encoder m=16 BART	49.1%	85.5%	95.3%	**3.17**	0.650
Furniture	Baseline	9.9%	24.7%	32.5%	36.3	0.177
	Bi-Encoder BART	**53.8%**	**86.9%**	**93.8%**	**3.36**	**0.679**
	Poly-Encoder m=16 BART	50.8%	85.5%	93.3%	3.46	0.657

Table 5: Response Retrieval Task performance on test-std set in fashion and furniture domains

Ground-truth action	Predicted action	Error Pct.
None	SearchDatabase	28.74%
SearchDatabase	SpecifyInfo	9.49%
None	SpecifyInfo	8.97%
SearchDatabase	None	7.02%
SearchMemory	SearchDatabase	6.89%

Table 6: Most frequent errors of the BART-large model for Sub-Task #1 in the fashion domain

Incorrectly predicted intents	Error Pct.
DA:REQUEST:GET:CLOTHING	10.13%
DA:INFORM:PREFER:CLOTHING	8.65%
DA:REQUEST:ADD_TO_CART:CLOTHING	5.76%
ERR:CHITCHAT	4.66%
DA:ASK:GET:CLOTHING.price	4.21%

Table 7: Most frequent errors of the BART-large model for Sub-Task #3 in the fashion domain

guage. And for Sub-Task #3, the F1 of intents and slots are measured to evaluate multimodal dialog state. The evaluation results of the BART models are the average of 5 experiment results, while those of the other models are the average of 2 experiment results due to limited time. The BART model shows the best performance and outperforms the baselines by large margins.

Our joint generation models do not consider action keywords as special tokens, but consider them as part of *normal* text, thus often splitting them into multiple tokens. For example, "AddToCart" is represented as a continuous token sequence of 'Add', 'To' and 'Cart'. Consequently, the models do not generate probabilities for the action candidates. We instead compute the probability of an action keyword for estimating action perplexity as follows: We employ beam search for the decoders of the models and collect the probabilities of the tokens of all action keywords. Then we apply softmax on the extracted token probabilities. Consider an action keyword (a_i) with K number of tokens (t_i^k for $k = 1, \cdots, K$). We compute the probability of the action keyword ($p(a_i)$) as follows:

$$\prod_{k=1}^{K} p\left(t_i^k | t_i^1, \cdots, t_i^{k-1}\right) \tag{11}$$

Like the baseline model, we normalize the probabilities of action keywords by applying softmax followed by logarithm on them.

Table 2 summarizes the average evaluation results of our best joint generation models and the best baseline models against the SIMMC test-std dataset in the fashion and furniture domains. Note that the SIMMC organizers did not release the ground-truth of the test-std dataset. The performance of our best models on the test-std dataset is close to that on the devtest dataset, which shows the robustness of our models. Our best models outperform the best baseline models on the test-std dataset for most of the evaluation metrics except action perplexity. The high action perplexity of our joint generation models can be due to the fact that they are not optimized for the action classification task. Also, our best models show slightly lower (1%) performance of action accuracy in the furniture domain than the best baseline models, as our best models show slightly higher (0.5%) performance for the devtest dataset in the furniture domain than the best baseline models.

Table 3 summarizes the evaluation results of our encoder-decoder models against the SIMMC devtest dataset in fashion domain, when the models are trained for the three generation subtasks individually or jointly. The results show that the joint learning models perform better than individual learning models.

Response Retrieval Task

We performed a variety of experiments to test two fine-tuned Transformer models (BERT, BART) to train the two response retrieval architectures (Bi-encoder, Poly-encoder), with different values of m for the poly-encoder. We trained the architectures and fine-tuned the models for 10 epochs with a batch size of 32 and a learning rate of $5e^{-5}$ for all the subsequent experiments. For each model, we conducted at least 5 experiments and the results were averaged.

Both Bi-encoder and Poly-encoder architectures based on BERT or BART models outperformed the baseline models for both fashion and furniture domains by wide margins. For the fashion domain, the results using the BART model were significantly higher than those using the BERT model, while the results of the two models in the furniture domain were close to each other. We performed further experiments with BART for the Poly-encoder architecture by varying the value of m. For the fashion domain, we found that the Bi-encoder architecture performs slightly better than the Poly-encoder architecture across several values of m, which is

User utterance		That's okay, thank you for looking. What about the polka dot one from earlier?
Ground truth	Action	None
	DST	DA:REQUEST:GET:CLOTHING [fashion-O=OBJECT_2, fashion-pattern=polka dot, fashion-sequential=earlier]
	System utterance	That one is only available in XL at this time.
Prediction	Action	SearchDatabase [info]
	DST	DA:ASK:GET:CLOTHING.info [fashion-O=OBJECT_1, fashion-pattern=polka dot, fashion-theme=one, fashion-SITUATION_1=earlier]
	System utterance	This is from Downtown Consignment and is available in size M. It 212 Localts $111 and is available in a medium.
Dialogue history		System: No problem! I've added an XL of that sweater to your cart. Would you like help with anything else today?
User utterance		No, that's all I need today. Thanks!
Ground-truth response		Have a great day!
Predicted response		You're welcome! Have a wonderful day!

Table 8: Examples of incorrect predictions by the joint generation (first example) and response retrieval (second example) models. Errors are colored in red.

in contrast to other observations (Humeau et al. 2020). We were able to achieve an average recall@1 score of 53.3%, a mean rank of 2.81, and an MRR of 0.681. For the furniture domain, the poly-encoder model with m=16 produced the best results overall, with a recall@1 of 48.4%, a mean rank of 3.39, and an MRR of 0.639. We compared the results of the fine-tuned BART models with the pre-trained BART without fine-tuning, and observed that the fine-tuning on the joint learning tasks improved the scores significantly, especially for the fashion domain. All the evaluation scores for the devtest dataset are shown in Table 4.

Table 5 summarizes the evaluation results of the Bi-encoder and Poly-encoder models with BART against the test-std dataset. Both our models outperform the best baseline models by large margins, and show similar performance in comparison to their performance on the devtest dataset, proving the robustness of our models. Bi-encoder models produce better results than Poly-encoder models for both fashion and furniture domains.

Error Analysis

In this section, we performed an error analysis of our joint generation BART-large model in the fashion domain. Table 6 shows the most frequent errors of our BART-large model in Sub-Task #1, where the predicted action is different from the ground-truth action. The model often incorrectly predicts non-empty actions when the ground-truth actions are empty (None). For the first example in Table 8, the model confuses "None" action with "SearchDatabase" action, and also produces invalid intent format. Table 7 shows the most frequent ground-truth intents in Sub-Task #3 that the BART-large model did not predict correctly.

We also performed an error analysis of the Bi-encoder on the fashion domain response retrieval and observed that there are two main types of errors. The first type of errors occurs when there are multiple similar candidate responses of a greeting or chit-chat nature, usually at the end of dialogues. The second example in Table 8 illustrates one such case. The "incorrect" predicted response is actually a reasonable response to the context. The second type of errors arises when the response is related to a product attribute (e.g. price, rating, availability), as our current models do not incorporate the product attribute knowledge base when selecting responses. As a result, the model picks a response with incorrect information.

5 Conclusion

We developed an end-to-end encoder-decoder model based on BART for generating outputs of the three tasks (Sub-Task #1, Sub-Task #2 Response, Sub-Task #3) in a single string, called the "joint generation" model, and another model based on Bi-Encoder and Poly-Encoder for generating outputs of the Sub-Task #2 Retrieval task, called "response retrieval" model. The two models are trained and evaluated separately. We achieved the first place in Sub-task #1 and Sub-Task #2 Retrieval categories, and the second place (runner-up) in Sub-Task #2 Generation and Sub-Task #3 categories, with the first place of overall ranking in Track 4 of the 9th Dialog State Tracking Challenge (DSTC 9) – SIMMC: Situated Interactive MultiModal Conversations. In particular, our response retrieval model significantly outperformed the other entries of the SIMMC official evaluation in the Sub-Task #2 Retrieval category.

Acknowledgements

This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

References

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Lawrence Zitnick, C.; and Parikh, D. 2015. VQA: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, 2425–2433.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. In *NeurIPS*.

Crook, P. A.; Poddar, S.; De, A.; Shafi, S.; Whitney, D.; Geramifard, A.; and Subba, R. 2019. SIMMC: Situated Interactive Multi-Modal Conversational Data Collection And Evaluation Platform. In *ASRU*.

Das, A.; Kottur, S.; Gupta, K.; Singh, A.; Yadav, D.; Moura, J. M.; Parikh, D.; and Batra, D. 2017. Visual dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 326–335.

De Vries, H.; Strub, F.; Chandar, S.; Pietquin, O.; Larochelle, H.; and Courville, A. 2017. Guesswhat?! visual object discovery through multi-modal dialogue. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5503–5512.

Dinan, E.; Roller, S.; Shuster, K.; Fan, A.; Auli, M.; and Weston, J. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *ICLR*.

Gunasekara, C.; Kim, S.; D'Haro, L. F.; Rastogi, A.; Chen, Y.-N.; Eric, M.; Hedayatnia, B.; Gopalakrishnan, K.; Liu, Y.; Huang, C.-W.; et al. 2020. Overview of the Ninth Dialog System Technology Challenge: DSTC9. *arXiv preprint arXiv:2011.06486* .

Guo, X.; Wu, H.; Cheng, Y.; Rennie, S.; Tesauro, G.; and Feris, R. 2018. Dialog-based interactive image retrieval. In *Advances in neural information processing systems*, 678–688.

Hosseini-Asl, E.; McCann, B.; Wu, C.-S.; Yavuz, S.; and Socher, R. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796* .

Humeau, S.; Shuster, K.; Lachaux, M.-A.; and Weston, J. 2020. Poly-encoders: Transformer Architectures and Pre-training Strategies for Fast and Accurate Multi-sentence Scoring. In *ICLR*.

Kadlec, R.; Schmid, M.; and Kleindienst, J. 2015. Improved deep learning baselines for Ubuntu corpus dialogs. In *Machine Learning for SLU & Interaction NIPS 2015 Workshop*.

Kottur, S.; Moura, J. M.; Parikh, D.; Batra, D.; and Rohrbach, M. 2019. CLEVR-Dialog: A diagnostic dataset for multi-round reasoning in visual dialog. In *NAACL*.

Le, H.; Sahoo, D.; Chen, N. F.; and Hoi, S. C. 2019. Multimodal transformer networks for end-to-end video-grounded dialogue systems. In *ACL*, 5612–5623.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 7871–7880.

Lowe, R.; Pow, N.; Serban, I.; and Pineau, J. 2015. The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 285–294.

Moon, S.; Kottur, S.; Crook, P. A.; De, A.; Poddar, S.; Levin, T.; Whitney, D.; Difranco, D.; Beirami, A.; Cho, E.; et al. 2020. Situated and Interactive Multimodal Conversations. *arXiv preprint arXiv:2006.01460* .

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* .

Rothe, S.; Narayan, S.; and Severyn, A. 2020. Leveraging Pre-trained Checkpoints for Sequence Generation Tasks. *Transactions of the Association for Computational Linguistics* 8: 264–280.

Saha, A.; Khapra, M.; and Sankaranarayanan, K. 2018. Towards building large scale multimodal domain-aware conversation systems. In *AAAI*, 696–704.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, 3776–3783.

Sukhbaatar, S.; Weston, J.; Fergus, R.; et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, 2440–2448.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Zhang, S.; Dinan, E.; Urbanek, J.; Szlam, A.; Kiela, D.; and Weston, J. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *ACL*, 2204–2213.