Evaluating and Improving Context Attention Distribution on Multi-Turn Response Generation using Self-Contained Distractions

Yujie Xing and Jon Atle Gulla

Norwegian University of Science and Technology

ABSTRACT

Despite the rapid progress of open-domain generation-based conversational agents, most deployed systems treat dialogue contexts as single-turns, while systems dealing with multi-turn contexts are less studied. There is a lack of a reliable metric for evaluating multi-turn modelling, as well as an effective solution for improving it. In this paper, we focus on an essential component of multi-turn generation-based conversational agents: context attention distribution, i.e. how systems distribute their attention on dialogue's context. For evaluation of this component, We introduce a novel attention-mechanism-based metric: DAS ratio. To improve performance on this component, we propose an optimization strategy that employs self-contained distractions. Our experiments on the Ubuntu chatlogs dataset show that models with comparable perplexity can be distinguished by their ability on context attention distribution. Our proposed optimization strategy improves both non-hierarchical and hierarchical models on the proposed metric by about 10% from baselines.

KEYWORDS

Natural Language Processing, Response Generation, Dialogue System, Conversational Agent, Multi-Turn Dialogue System

1. INTRODUCTION

In recent years, generation-based conversational agents have shown a lot of progress, while multiturn generation-based conversational agents are still facing challenges. Most recent work ignores multiturn modelling by considering a multi-turn context as a 1-turn context [1, 2]. Some works try to deal with multi-turn modelling using modified attention mechanisms, hierarchical structures, utterance tokens, etc. [3, 4, 5]. The main difference between multi-turn conversational agents and regular (1-turn) conversational agents is that instead of dealing with an utterance in a context on the word-level, multi-turn models deal with a dialogue on the utterance-level, so that models can understand an utterance as a whole and focus on important utterances rather than important words.

An example of important/unimportant utterances existing in the same context is given by Table 1.

Table 1: An example of important utterances and unimportant utterances under the same context in the Ubuntu chatlog dataset [6]. Unimportant utterances are marked in red.

User	Utterances
Taru	Haha sucker.
Kuja	?
Taru	Anyways, you made the changes right?
Kuja	Yes.
Taru	Then from the terminal type : sudo apt - get update
Kuja	I did.

In this example, the first two utterances ("Haha sucker." and "?") are unimportant utterances that are irrelevant to the main topic of the context. Human dialogues naturally contain many of these unimportant utterances. These utterances do not distract humans from understanding the main idea of the context, since humans can easily ignore them and focus instead on important utterances; however, a model usually lacks this capability and can be distracted by these utterances, resulting in a lower performance in generating relevant responses to the main topic of a context. Therefore, it is crucial that a multi-turn model can decide which utterances in the context are important and which are unimportant, and distribute its attention accordingly. In this paper, we define the research topic as context attention distribution, which denotes how much attention is distributed respectively to important and unimportant utterances in a context. A model with a good performance on context attention distribution should pay more attention to important utterances and less attention to unimportant utterances.

Recent work lacks a measurement for the performance of multi-turn modelling. Common metrics rely on general evaluation metrics such as BLEU [7], which measures the quality of generated responses. These metrics cannot directly describe a model's ability on dealing with multi-turn contexts, since the quality of generated responses is influenced by many aspects. Better performance in dealing with multi-turn context may result in better general performance; however, a better general performance does not necessariy mean that the model has a better ability on dealing with multi-turn contexts. Thus, as a supplementary to general evaluation metrics like BLEU, we propose a metric that measures a conversational agent's performance on context attention distribution, which is specifically designed for evaluating a model's performance on multi-turn modelling. Since most multi-turn conversational agents have the attention mechanism and rely on it to distribute attention to different utterances in a context, we propose distracting test as the evaluation method to examine if a model pays more attention to the important utterances. The test adds unrelated utterances as distractions to the context of each dialogue and compares the attention scores of distracting utterances (i.e., unimportant utterances) and original utterances (i.e., important utterances). The ratio of the average attention score of distracting utterances and original utterances is defined as the distracting attention score ratio (DAS ratio). We use DAS ratio as the evaluation metric for a model's performance on context attention distribution. A model with good capability on context attention distribution should have higher scores on original utterances and lower scores on distracting utterances, thus a lower DAS ratio.

Furthermore, we propose a self-contained optimization strategy to improve a conversational agent's performance on context attention distribution. For each dialogue, we randomly pick some utterances from the training corpus outside the current dialogue as self-contained distractions, and insert them into the current dialogue with different levels of possibilities. The attention paid to these distractions is minimized during the training process through multi-task learning. With this

optimization strategy, a model learns to distribute less attention to unimportant utterances and thus more attention to important utterances.

In this paper, we examine the following research questions: 1) How do existing multi-turn modeling structures perform on context attention distribution? 2) Can the proposed optimization strategy improve a model's performance on context attention distribution? 3) Which probability level is the best for inserting distractions in the proposed optimization strategy?

Our contributions are as follows:

(1) We deal with a less studied problem: evaluating and improving context attention distribution for multi-turn conversational agents.
(2) We propose a novel evaluation metric for multi-turn conversational agents: DAS ratio. It measures a model's performance on context attention distribution, i.e. the capability of distributing more attention to important utterances and less to unimportant ones.
(3) We propose an optimization strategy that minimizes the attention paid to self-contained distractions during the training process, and thus makes the model try to pay less attention to unimportant utterances. The strategy can easily be added and adapted to existing models.

Extensive experiments on 23 model variants and 9 distracting test sets show an overall improvement in the performance on context attention distribution for the proposed strategy. We will share our code for reproducibility (in the final version, a Github link will be provided). Related work is introduced in Section 2. In Section 3, we introduce our base models and proposed methods. We show our experiments settings in Section 4 and results in Section 5. Finally, we give a conclusion in Section 6.

2. RELATED WORKS

Common evaluation metrics for conversational agents measure the similarity between the generated responses and the gold responses. Liu et al. [8] summarizes commonly used metrics: word overlap-based metrics (e.g. BLEU) and embedding-based metrics. Bruni et al. [9] propose an adversarial evaluation method, which uses a classifier to distinguish human responses from generated responses. Lowe et al. [10] propose a model that simulates human scoring for generated responses. Zemlyanskiy et al. [11] examine the quality of generated responses in a different direction: how much information the speakers exchange with each other. Recently, Li et al. [5] propose a metric that evaluates the human-likeness of the generated response by measuring the gap between the corresponding semantic influences. Different from the above, our proposed evaluation metric is based on the attention mechanism and is intended to measure a model's performance on context attention distribution.

Most generation-based conversational agents apply simple concatenation for multi-turn conversation modelling [2, 1], which regards a multi-turn context as a 1-turn utterance. Some works try to model multi-turn conversations through the hierarchical structure: Serban et al. [3, 4] first introduce the hierarchical structure to dialogue models. Tian et al. [12] evaluate different methods for integrating context utterances in hierarchical structures. Zhang et al. [13] further evaluate the effectiveness of static and dynamic attention mechanism. Gu et al. [14] apply a similar hierarchical structure on Transformer, and propose masked utterance regression and distributed utterance order ranking for the training objectives. Different from hierarchical models, Li et al. [5] encode each utterance with a special token [C] and apply a flow module to train the model to predict the next [C]; then they use semantic influence (the difference of the predicted and original tokens) to support generation. In our paper, instead of modelling the relations of

inter-context utterances as [14] or the dialogue flow as [5], our optimization strategy improves multi-turn modelling by n distinguishing important/unimportant utterances directly on the attention mechanism.

3. METHODS

Our proposed evaluation metric and optimization strategy can work on attention mechanisms including Transformers. In this paper, we choose an LSTM Seq2Seq model with attention mechanism [15, 16, 17] as the base model, since most hierarchical structured multi-turn conversational agents are based on LSTM [3, 4, 12, 13] while few are based on Transformers.

The basic task of generation-based conversational agents is to predict the next token given all the past and current tokens from the context and response, and to make the predicted response as similar to the original response as possible. Formally, the probability of response Y given context X is predicted as:

$$P(Y|X) = \prod_{t=1}^{n} p(y_t|y_1, \ldots, y_{t-1}, X), \qquad (1)$$

Where $X = x_1, \ldots, x_m$ and $Y = y_1, \ldots, y_n$ are a context-response pair.

3.1. LSTM Seq2Seq Model with Attention

We simplify an LSTM unit as *LSTM*, and we denote the attention version of an LSTM with an asterisk (*LSTM**). They are well introduced in previous work [18].We calculate the hidden vector h_t at step t as:

$$h_t = LSTM^*(h_{t-1}, E(z_t), c_{t-1}), \qquad (2)$$

where $h_{t-1} \in R^{dim}$ is the hidden vector at step t-1, dim is the dimensionality of hidden vectors, and $E(z_t)$ is the word embedding for token $z_t \in \{x_1, \ldots, x_m, y_1, \ldots, y_{n-1}\}$. c_{t-1} is the context vector at step t-1, and it is input to the next step t *only in the decoder*. Each h_t and c_t of the current step t are combined through a linear layer and an activation to predict the next token.

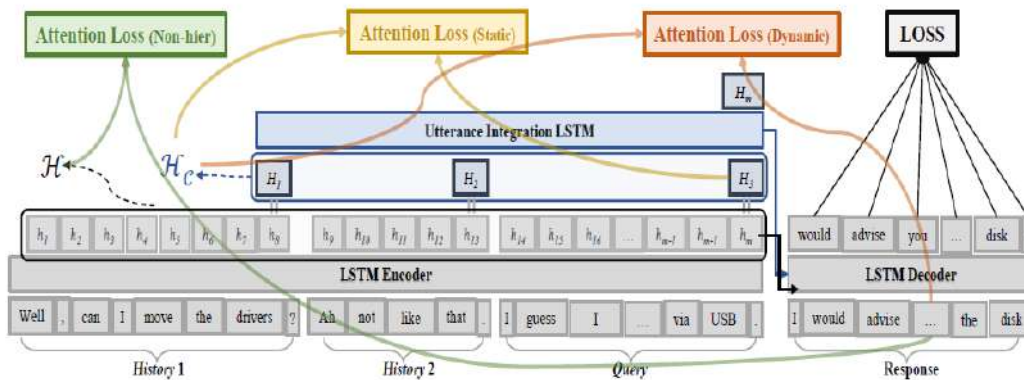### 3.2. Attention Mechanism & Utterance Integration (UI)

Figure 1: Structure of non-hierarchical, static and dynamic attention loss.

We examine both non-hierarchical and hierarchical structures. For hierarchical structures, following [13], we develop two attention mechanisms: static and dynamic. Following [12], we develop models that are both with and without utterance integration LSTM units. For the non-hierarchical structured model, there are no hidden vectors for utterances. All hidden vectors of tokens in the encoder are concatenated and used in the attention mechanism. Denoting the concatenated vector $H = [h_1, h_2, ..., h_m]$, we calculate the context vector c_t for each decoding
Step t as:

$$c_t = H \cdot (\text{softmax}(H \cdot h_t)) . \tag{3}$$

For the hierarchical models, we use the hidden vector of each utterance's last token as the hidden vector of the utterance, and we discard the hidden vectors for the other tokens. Thus, compared to the non-hierarchical structured model, we have much fewer hidden vectors from the encoder.

The context vector of static attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of the last utterance in the context. Denoting the hidden vector of k th utterance as H_k, and the hidden vector of the last utterance in the context as H_q, we have the context's concatenated vector $H_C = [H_1, H_2, ..., H_q]$. We calculate the context vector c_t for static attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot H_q)), \tag{4}$$

where it is easy to see that the static context vector remains unchanged by the decoder.

The context vector of dynamic attention mechanism is calculated based on the utterance-level concatenated vector and the hidden vector of each token in the decoding step. We calculate the context vector c_t for dynamic attention mechanism as:

$$c_t = \mathcal{H}_\mathcal{C} \cdot (\text{softmax}(\mathcal{H}_\mathcal{C}^\top \cdot h_t)) . \tag{5}$$

Compared to the static attention mechanism, the context vector c_t varies at each decoding step. Finally, with the utterance integration LSTM unit, we calculate H_m from $H_1, H_2, ... H_q$:

$$H_m = LSTM(H_1, H_2, ..., H_q). \tag{6}$$

For models with utterance integration (UI), H_m is input to the first step of the decoder, while for models without UI, regular h_m is input instead.

3.3. Distracting Test & Attention Score (AS)

We examine if a multi-turn conversational agent distributes more attention to important utterances through the distracting test and attention scores.

In the distracting test, for each dialogue before the end of the context, we insert several utterances that are irrelevant to the main idea of the dialogue as distractions. These utterances are named *distracting utterances*, and they can be randomly picked utterances from the training corpus (**random**), be formed by frequent words from the training corpus (**frequent**), or be formed by rare words from the training corpus (**rare**). We compare the attention scores of the distracting utterances with the attention scores of the original utterances. A well-performing model should distribute less attention to the distracting utterances while more attention to the original utterances. For an utterance H_k, the corresponding attention score $AS(H_k)$ is calculated as:

$$AS(H_k) = \begin{cases} \dfrac{m}{q} \cdot \text{mean}_t \left(\dfrac{\sum_{h_i \in H_k} \exp(h_i^\top \cdot h_t)}{\sum_{i=1}^{m} \exp(h_i^\top \cdot h_t)} \right) & \text{Non-hierarchical} \\[2ex] \dfrac{q \cdot \exp(H_k^\top \cdot H_q)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot H_q)} & \text{Static attention} \\[2ex] \text{mean}_t \left(\dfrac{q \cdot \exp(H_k^\top \cdot h_t)}{\sum_{k=1}^{q} \exp(H_k^\top \cdot h_t)} \right) & \text{Dynamic attention} \end{cases} \tag{7}$$

h_i denotes hidden vectors from the encoding steps and h_t denotes hidden vectors from the decoding steps. m is the number of tokens in a context, and q denotes the number of utterances in a context. Note that for non-hierarchical models we multiply by an m in each $AS(H_k)$ to avoid bias caused by the total number of tokens in different contexts. Similarly for hierarchical models, we multiply by a q in each $AS(H_k)$ to avoid bias caused by the number of total utterances in different contexts. As a result, for an utterance H_q, $AS(H_q)$ will be 100% (or approximately 100% for non-hierarchical models) if the model assigns H_q an about average attention score among all utterances. We denote the last utterance in a context as *Query* and the rest of utterances in the context as *History*. Since different models have different scalars on attention scores, we calculate the average AS for all distracting utterances and all *History* in each dialogue, and use the ratio of them for evaluation. This ratio is denoted as distracting attention score ratio (**DASratio**),which measures a model's ability on context attention distribution:

$$\text{DAS ratio} = \text{mean}_{d \in D} \left(\frac{\text{mean}(AS(H_{\text{Distraction}}))}{\text{mean}(AS(H_{History}))} \right), \tag{8}$$

where d means a single dialogue, and D denotes all dialogues in a test set. $H_{\text{Distraction}}$ denotes distracting utterances, and $H_{History}$ denotes utterances in *History*.

3.4. Optimization with Self-Contained Distractions on Attention Mechanism

To train a conversational model to distribute more attention to important and less attention to unimportant utterances, we propose the following optimization strategy:1) For each dialogue, we select some random utterances from other dialogues in the training corpus as self-contained distractions. We decide whether to insert these distractions into the current dialogue or not stochastically by a probability level. We denote the probability level as the training inserting probability. The locations of inserting distractions are randomly decided, while the locations are always before Query (the last utterance of the context). 2) We create a bitmask M to track whether an utterance is original (0) or distracting (1). During the training period, the model uses the bitmask to calculate the attention loss $L^t_{\text{attention}}$, which is summed up with the loss from the response generator. For each decoding step t, the attention loss is calculated as:

$$\mathcal{L}^t_{attention} = \begin{cases} \text{MSE}(\text{softmax}(\mathcal{H}^\top \cdot h_t) \circ M, 0) & \text{Non-hierarchical} \\ \text{MSE}(\text{softmax}(\mathcal{H}_C^\top \cdot H_q) \circ M, 0) & \text{Static attention} \\ \text{MSE}(\text{softmax}(\mathcal{H}_C^\top \cdot h_t) \circ M, 0) & \text{Dynamic attention} \end{cases} \tag{9}$$

where $\circ$ means Hadamard product, or element wise multiplication. As shown in Equation (9), our goal is to minimize the attention assigned to all the self-contained distractions. During the distracting test, no bitmask is offered to the model. The illustration of attention loss on both non-hierarchical and hierarchical models is shown in Figure 1.

4. EXPERIMENTS

In this section, we introduce the setups of the experiment.

4.1. Dataset

We use the Ubuntu chatlogs data set [6] as the training and testing corpus, which contains dialogues about solving technical problems of Ubuntu. We choose this dataset because the dialogues have both technical topics and casual chats, meaning that it is easier to distinguish important/unimportant utterances than datasets whose topics are consistent. We use about $0.48M$ dialogues for training, $20K$ dialogues for validation, and $10K$ dialogues for testing. These are the original settings of the Ubuntu chatlogs dataset. We removed all single-turn dialogues.

4.2. Training

Our methods are built on an LSTM Seq2Seq model with attention mechanism. We used Pytorch [19] for implementation. The LSTM model has 4 layers and the dimension is 512. The training procedure was with a batch size of 256, a learning rate of 1.0, and a gradient clip threshold of 5. The vocabulary size is 25000 and the dropout rate is 0.2. The learning rate is halved when the perplexity stops dropping, and the training is stopped when the model converges.

4.3. Examined Models

We examine our proposed evaluation metric on 5 models: non-hierarchical LSTM (Non-hier), static attention without utterance integration LSTM unit (Static), static attention with utterance integration LSTM unit (StaticUI), dynamic attention without utterance integration LSTM unit (Dynamic), and dynamic attention with utterance integration LSTM unit (DynamicUI). In addition, we examine our proposed optimization strategy on these 5 models with 3 training inserting probabilities–0.5, 0.7, and 1.0. Models with a training inserting probability of 0 are regarded as baselines. For comparison, we pick the best overall model and train the model with self-contained distractions but without training on the attention loss (Non-atten-loss), i.e. the model does not know which utterances are distractions. In total, we train and evaluate 23 model variants.

4.4. Evaluation

Table 2: Examples of distracting test sets. Distracting utterances are marked red.

	Random: 0.5	Random: 0.7	Random: 1.0
History	\	Well, can I move the drives?	Yes.
	Or kill all speedlink.	Anyways, you made the Changes right?	Well, can I move the drives?
	Well, can I move the drives?	Ah not like that.	Then from the terminal type: sudoapt-get update.
	Ah not like that.	I did.	Ah not like that.
	Frequent: Begin	**Frequent: Middle**	**Frequent:End**
History	Why should I help you?	Well, can I move the drives?	Well, can I move the drives?
	I have my right.	Why should I help you?	Ah not like that.
	Well, can I move the drives?	I have my right.	Why should I help you?
	Ah not like that.	Ah not like that.	I have my right.
	Rare:Begin	**Rare:Middle**	**Rare:End**
History	Would you have lunch?	Well, can I move the drives?	Well, can I move the drives?
	I should have lunch.	Would you have lunch?	Ah not like that.
	Well, can I move the drives?	I should have lunch.	Would you have lunch?
	Ah not like that.	Ah not like that.	I should have lunch.
Query	**I guess I could just get an enclosure and copy via USB.**		
Response	**I would advise you to get the disk.**		

For the distracting test, we set the number of distracting utterances for each dialogue to 2. We chose 2 to make the distracting utterances a complete turn and to make the number of distracting utterances the minimum, since dialogues from the corpus normally have only 4 to 8 utterances in the contexts. We have 3 distracting test sets. 1) Random distracting test set: distracting utterances in this test set are randomly picked from the training corpus (outside the current dialogue), and they are randomly picked in every evaluation step, which means that there is no pre-prepared random distracting test set. 2) Frequent distracting test set: distracting utterances in this test set are formed by frequent words in the training corpus, but these utterances do not appear in the training corpus. In our experiments, we use "why should I help you" and "I have my right" as examples of distracting utterances with frequent words. 3) Rare distracting test set: distracting utterances in this test set have words that are rare in the training corpus, and these utterances do not appear in the training corpus. In our experiments, we use "would you have lunch?" and "I should have lunch" as examples of distracting utterances with rare words.

In the distracting test, we insert distracting utterances into different locations. For 1) random, we insert utterances to a random location before Query in each context. Similar to the optimization strategy, we use different probability levels to decide whether a distracting utterance is to be inserted or not. We denote these as testing inserting probability. In our experiments, we set the probability levels to be 0.5, 0.7, and 1.0. We expect the model to perform stably on all different probability levels. For 2) frequent and 3) rare, we have three kinds of inserting locations: at the beginning of a context (marked as Begin), in the middle of the context (marked as Middle), and at the end of the context (before Query and after History, marked as End). In total, we have 9 test sets for evaluation. See Table 2 for the example of each test set.

5. RESULTS AND DISCUSSIONS

Table 3 illustrates the main results on DAS ratios. It shows the DAS ratios of 23 trained model variants on 9 distracting test sets. Figure 2 shows the DAS ratios of 3 example model variants

(StaticUI with training inserting probability of 0.0 as the baseline, Non-atten-loss StaticUI with training inserting probability of 0.7, and StaticUI with training inserting probability of 0.7) on 9 distracting test sets. Table 4, Table 5 and Table 6 show the detailed results on average Attention Score (average AS) of distracting utterances and average AS of History.

In Table 3, we show the perplexity and History's average AS of each model on the non-distracted test set under the "Original" column. Since perplexity scores on the distracting test sets are similar, we show the perplexity scores on the non-distracted test set only. We show the DAS ratios of each model on each of the distracting test sets under the "DAS ratio for distracting test set" column. A lower DAS ratio means that a model distributes less attention to distracting utterances (unimportant utterances) and more attention to the original utterances in History (important utterances), from which it can be inferred that the model has better performance on context attention distribution. Both perplexity and DAS ratio are the lower, the better.

5.1. Perplexity and Average AS on Non-Distracted Test Set

Perplexity scores are shown in the "Perp." column, under the "Original" column in Table 3. Perplexity scores of the examined 23 models are similar; the Static models trained with our proposed optimization strategy and a higher training inserting probability level achieves slightly better performance than other models. Average AS are shown in the "Avg." column, under the "Original" column in Table 3. The average AS of History tells about a model's attention distribution for History and Query. A higher score indicates that less attention is distributed to Query. Recall that AS of an utterance is 100% (or approximately 100% for non-hierarchical models) if the utterance is paid about average attention among the dialogue. Overall, the models distribute attention of lower than average to History, especially for models with static attention (i.e. the Static model and StaticUI model), which distribute more attention to Query than non hierarchical models and models with dynamic attention.

This is apparent from the structure of static attention. We also show the results of a StaticUI model without training on the attention loss (Non-atten-loss StaticUI model) as a comparison. The StaticUI model trained with our optimization strategy distributes more attention to query than the Non-atten-loss StaticUI model. This is because the optimization strategy decreases the model's attention distributed to distracting utterances in History, thus decreasing the overall attention distributed to History.

5.2. Distracting Test: Random

Results of the random distracting test with different testing inserting probabilities (0.5, 0.7, and 1.0) are shown in the "Random" column in Table 3. Models with training inserting probabilities of 0.0 (shown in the row where "Prob" is 0.0) are baseline models to which our proposed optimization strategy is not applied. In general, our proposed optimization strategy with training inserting probabilities of 0.5 or 0.7 achieves better performance on DAS ratios (i.e. the models achieve lower DAS ratios) on random distracting test sets of all 3 testing inserting probabilities. The Static model and the DynamicUI model achieves the best performance with a training inserting probability of 0.5, while the Non-hier model, the StaticUI model and the Dynamic model achieve the best performance with a training inserting probability of 0.7. A training inserting probability
of 1.0 leads to worse performance. One reason is that it assumes there must be some distracting utterances in a context, while that is not always the case.

Table 3: Results of perplexity (Perp.) and average AS of History (Avg.) on the original test set (%) are shown in the "Original" column. We also show results of DAS ratios on 9 distracting test sets and 23

model variants.

| Prob	Model Structure	Original Perp.	Original Avg.	Random 0.5	Random 0.7	Random 1.0	Frequent Begin	Frequent Middle	Frequent End	Rare Begin	Rare Middle	Rare End
0.0	Non-hier	43.2	91.3	0.93	0.93	0.93	0.75	0.80	0.84	0.80	0.92	1.01
	Static	44.1	61.4	0.82	0.82	0.79	0.37	0.80	1.31	0.37	0.77	1.21
	StaticUI	44.6	57.5	0.79	0.76	0.76	**0.32**	0.75	1.32	0.30	0.75	1.22
	Dynamic	45.4	81.4	0.89	0.89	0.88	0.65	0.86	1.02	0.66	0.89	1.06
	DynamicUI	44.7	91.6	0.94	0.94	0.93	0.72	0.84	0.86	0.73	0.93	0.97
	Non-atten-loss StaticUI	44.7	71.1	0.73	0.73	0.72	0.39	0.68	0.93	0.40	0.80	1.11
0.5	Non-hier	43.4	87.2	0.84	0.83	0.81	0.63	0.74	**0.76**	0.69	0.81	0.86
	Static	44.5	66.5	0.70	0.69	0.67	0.42	0.78	1.12	0.34	0.71	0.99
	StaticUI	44.3	47.7	0.74	0.74	0.70	0.39	0.71	1.08	0.40	**0.69**	0.96
	Dynamic	44.6	81.9	0.79	0.78	0.77	0.64	0.74	0.84	0.61	0.77	0.85
	DynamicUI	43.9	86.7	0.82	0.81	0.80	0.60	0.84	0.87	0.61	0.80	0.83
0.7	Non-hier	43.2	86.9	0.84	0.82	0.80	0.72	0.82	0.82	0.71	0.85	0.87
	Static	**44.0**	57.6	0.73	0.72	0.69	0.40	0.70	1.08	0.41	0.70	0.98
	StaticUI	44.9	43.7	**0.67**	**0.67**	**0.65**	0.36	**0.66**	1.02	0.36	0.70	0.99
	Dynamic	44.3	82.0	0.76	0.75	0.73	0.58	0.71	0.86	0.58	0.73	0.83
	DynamicUI	44.8	85.3	0.93	0.93	0.93	0.45	0.78	0.80	0.60	0.80	**0.81**
	Non-atten-loss StaticUI	44.1	55.4	0.72	0.70	0.69	0.45	0.70	0.98	0.43	0.73	0.97
1.0	Non-hier	47.3	95.9	0.91	0.90	0.90	0.84	0.86	0.85	0.85	0.87	0.88
	Static	**44.0**	65.4	0.70	0.70	0.68	0.49	0.74	1.08	0.46	0.71	0.88
	StaticUI	49.6	73.5	0.96	0.95	0.94	0.66	0.86	1.53	**0.21**	0.86	1.50
	Dynamic	44.7	88.8	0.79	0.78	0.77	0.63	0.75	0.82	0.65	0.77	0.82
	DynamicUI	45.2	90.2	0.87	0.86	0.85	0.73	0.81	0.83	0.75	0.88	0.88
	Non-atten-loss StaticUI	44.1	76.5	0.72	0.71	0.69	0.49	0.74	0.98	0.49	0.77	0.98

DAS ratio on distracting test sets
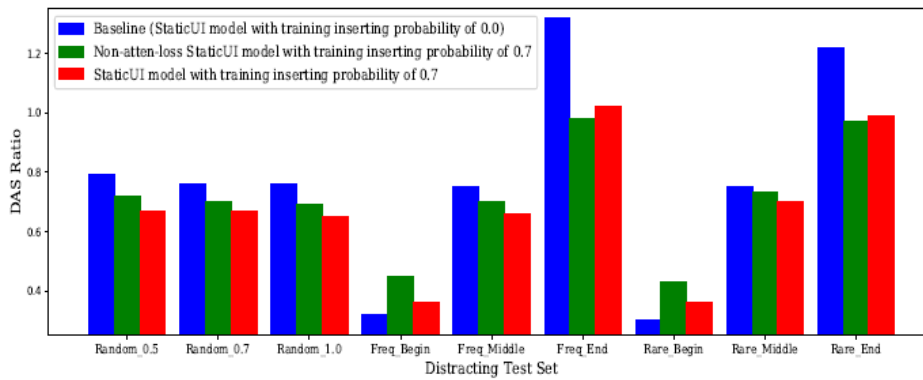
Figure 2: DAS ratios of 3 example model variants on 9 distracting test sets. The lower the DAS ratio, the better the performance.

The StaticUI model with a training inserting probability of 0.7 achieves the best overall performance on DAS ratio. As shown in Figure 2, on all the random distracting test sets (probabilities of 0.5, 0.7, and 1.0), the StaticUI model is better than the baseline StaticUI model and the Non-atten-loss StaticUI model. The baseline model is not trained with any self-contained distractions (training inserting probability is 0.0), and it gets the worst performance. The Non atten-loss model is trained with self-contained distractions (with a training inserting probability of 0.7) while not knowing which utterances are distractions, and it achieves a better performance than the baseline. The StaticUI model with a training inserting probability of 0.7 is trained to minimize the attention loss of self-contained distractions and it achieves the best performance. Naturally since the optimization strategy minimizes the attention loss of distractions, the StaticUI model distributes less attention to History and more attention to Query (refer to the "Avg" column in Appendix 4 for more details); nevertheless, a lower DAS ratio shows that the model distributes even less attention to the distracting utterances compared to the original utterances in History.

Note that even if both our proposed strategy and the random distracting test use the same trick: insert random distracting utterances among original utterances in History, the random utterances inserted in the distracting test are different from those inserted in the training process, thus it is difficult for the test to be biased in favor of models with our proposed strategy. It is apparent that less attention is distributed to History, while DAS ratio calculates the ratio between the distracting utterances and the original utterances in History, so it shows the attention distributed to the distracting utterances regardless of the total attention distributed to History. Moreover, we adopt three testing inserting probability levels to ensure stable evaluation results for each model.

5.3. Distracting Test: Frequent and Rare

Results of the frequent and the rare distracting test are shown in the "Frequent" and "Rare" columns in Table 3. Different from the random distracting test, the inserting locations of these two tests are decided manually. As a nature of LSTM model, all models distribute more attention to utterances near Query and less attention to utterances far away from Query, as can be seen in Table 3 and Figure 2 that DAS ratios are higher for End test set (near Query) and lower for Begin test set (far away from Query). Since the results on Begin and End test sets are biased by the structure of LSTM, we mainly analyze the results on Middle test sets. For the Middle test sets of both the frequent and rare distracting test, the best models are still those trained with our proposed optimization strategy. StaticUI models with training inserting probabilities of 0.5 and 0.7 achieve the best performance (lowest DAS ratios) on the Frequent Middle and Rare Middle

test sets. The Non-atten-loss models can be better than the models trained with a wrong training inserting probability. Telling from similar DAS ratios, the frequent distracting test set is as difficult for the trained models to distinguish as the rare distracting test set, although for humans, the rare distracting utterances are much easier to distinguish than the frequent ones.

Table 4: Results of perplexity (Perp.) and average AS of History (Avg.) on the original test set (%) are shown in the "Original" column. Besides, we show the results on the random distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), and average AS of original utterances in History (Avg.) (%).

Model		Original		Distracting Test Set								
Probability	Structure	Perp.	Avg.	Random 0.5			Random 0.7			Random 1.0		
				DAS ratio	DAS	Avg.	DAS ratio	DAS	Avg.	DAS ratio	DAS	Avg.
0.0	Non-hier	43.2	91.3	0.93	86.8	93.1	0.93	87.1	93.8	0.93	87.6	94.5
	Static	44.1	61.4	0.82	52.6	64.5	0.82	53.5	65.6	0.79	53.4	67.6
	StaticUI	44.6	57.5	0.79	47.4	60.2	0.76	46.3	61.3	0.76	47.7	62.7
	Dynamic	45.4	81.4	0.89	74.9	84.4	0.89	75.6	85.2	0.88	76.2	86.6
	DynamicUI	44.7	91.6	0.94	87.5	93.4	0.94	88.0	93.7	0.93	88.2	94.5
0.5	Non-hier	43.4	87.2	0.84	77.2	91.6	0.83	77.1	93.3	0.81	77.0	95.5
	Static	44.5	66.5	0.70	50.3	71.5	0.69	50.5	73.5	0.67	51.1	76.5
	StaticUI	44.3	47.7	0.74	38.1	51.2	0.74	39.2	53.1	0.70	39.1	55.5
	Dynamic	44.6	81.9	0.79	68.3	86.6	0.78	69.1	88.2	0.77	69.4	90.8
	DynamicUI	43.9	86.7	0.82	74.5	91.1	0.81	75.2	92.5	0.80	75.8	94.8
	Non-labled StaticUI	44.7	71.1	0.73	55.5	75.6	0.73	56.6	77.1	0.72	57.4	79.6
0.7	Non-hier	43.2	86.9	0.84	76.5	91.3	0.82	75.9	93.1	0.80	75.9	95.4
	Static	**44.0**	57.6	0.73	45.5	62.2	0.72	45.8	64.0	0.69	46.4	66.9
	StaticUI	44.9	**43.7**	**0.67**	**32.4**	**48.1**	**0.67**	**33.2**	**49.9**	**0.65**	**34.0**	**52.1**
	Dynamic	44.3	82.0	0.76	66.3	87.2	0.75	66.6	89.1	0.73	67.2	91.8
	DynamicUI	44.8	85.3	0.93	86.8	93.1	0.93	87.1	93.8	0.93	87.6	94.5
	Non-labled StaticUI	44.1	55.4	0.72	43.3	59.9	0.70	43.4	62.0	0.69	44.3	64.4
1.0	Non-hier	47.3	95.9	0.91	88.7	98.0	0.90	89.3	98.7	0.90	89.5	99.9
	Static	**44.0**	65.4	0.70	49.7	71.1	0.70	51.3	73.1	0.68	51.8	76.4
	StaticUI	49.6	73.5	0.96	74.8	77.8	0.95	75.2	79.4	0.94	76.7	81.2
	Dynamic	44.7	88.8	0.79	74.2	93.4	0.78	74.4	95.2	0.77	75.4	97.4
	DynamicUI	45.2	90.2	0.87	81.3	93.6	0.86	81.5	94.9	0.85	81.9	96.5
	Non-labled StaticUI	44.1	76.5	0.72	59.5	82.1	0.71	59.7	84.4	0.69	60.3	87.6

Table 5: Results on the frequent distracting test of: DAS ratio, average AS of distracting utterances(DAS) (%), average AS of original utterances in History (Avg.) (%), and AS of the first/last utterance in History (%).

Probability	Structure	Distracting Test Set										
		Frequent: Begin				Frequent: Middle			Frequent: End			
		DAS ratio	DAS	Avg.	1st	DAS ratio	DAS	Avg.	DAS ratio	DAS	Avg.	Last
0.0	Non-hier	0.75	70.8	94.1	82.3	0.80	74.7	93.2	0.84	78.0	92.7	98.4
	Static	0.37	29.3	79.5	47.9	0.80	56.7	71.2	1.31	82.0	62.4	72.6
	StaticUI	**0.32**	**24.3**	**76.1**	**42.2**	0.75	50.6	67.7	1.32	77.2	58.7	69.5
	Dynamic	0.65	60.9	93.3	72.7	0.86	75.5	88.0	1.02	87.5	85.1	88.8
	DynamicUI	0.72	72.2	100.7	81.0	0.84	81.6	97.0	0.86	84.1	98.2	98.8
0.5	Non-hier	0.63	57.3	91.0	84.0	0.74	66.4	89.7	**0.76**	68.7	90.8	99.5
	Static	0.42	35.0	84.5	47.4	0.78	59.8	77.0	1.12	80.3	71.5	81.7
	StaticUI	0.39	24.6	62.9	44.3	0.71	41.8	58.9	1.08	**54.7**	**50.6**	**55.2**
	Dynamic	0.64	60.4	94.6	76.7	0.74	68.2	92.1	0.84	76.2	91.1	94.4
	DynamicUI	0.60	60.1	100.4	77.7	0.84	78.4	92.9	0.87	82.3	94.2	93.0
	Non-labled StaticUI	0.39	35.1	90.3	53.7	0.68	56.1	82.7	0.93	72.6	78.4	91.2
0.7	Non-hier	0.72	64.7	90.1	83.1	0.82	72.6	88.8	0.82	73.4	89.4	97.1
	Static	0.40	30.3	75.2	48.3	0.70	48.7	69.3	1.08	68.1	62.9	69.2
	StaticUI	0.36	21.0	57.6	37.4	**0.66**	**36.0**	**54.7**	1.02	51.7	50.6	56.4
	Dynamic	0.58	56.3	96.8	73.9	0.71	66.4	93.4	0.86	76.0	88.8	91.8
	DynamicUI	0.45	44.4	98.8	76.1	0.78	73.2	93.8	0.80	75.8	95.2	95.0
	Non-labled StaticUI	0.45	31.7	70.6	51.0	0.70	46.6	66.6	0.98	60.2	61.2	65.8
1.0	Non-hier	0.84	82.0	97.8	92.9	0.86	83.7	97.7	0.85	83.2	97.4	100.0
	Static	0.49	40.2	82.5	60.3	0.74	57.1	76.8	1.08	73.3	67.8	72.3
	StaticUI	0.66	73.3	110.4	24.9	0.86	71.5	82.9	1.53	104.8	68.3	88.6
	Dynamic	0.63	64.0	102.1	81.7	0.75	73.9	98.5	0.82	79.5	97.5	99.7
	DynamicUI	0.73	72.5	100.0	83.5	0.81	79.2	97.4	0.83	81.6	98.7	97.6
	Non-labled StaticUI	0.49	46.1	95.0	67.1	0.74	65.0	87.7	0.98	79.9	81.6	86.6

Computer Science & Information Technology (CS & IT)

Table 6: Results on the rare distracting test of: DAS ratio, average AS of distracting utterances (DAS) (%), average AS of original utterances in History (Avg.) (%), and AS of the first/last utterance in History (%).

Model			Distracting Test Set										
Probability	Structure	Rare: Begin				Rare: Middle			Rare: End			
		DAS ratio	DAS	Avg.	1st	DAS ratio	DAS	Avg.	DAS ratio	DAS	Avg.	Last
0.0	Non-hier	0.80	74.7	93.8	82.3	0.92	84.3	92.1	1.01	92.0	91.2	96.7
	Static	0.37	29.3	79.6	47.9	0.77	55.5	72.2	1.21	77.8	64.3	75.2
	StaticUI	0.30	22.9	76.4	42.2	0.75	51.1	67.9	1.22	74.2	61.0	73.2
	Dynamic	0.66	61.4	93.6	72.7	0.89	77.4	87.2	1.06	87.0	82.3	85.4
	DynamicUI	0.73	73.8	100.5	81.0	0.93	87.9	94.2	0.97	90.1	93.0	93.0
0.5	Non-hier	0.69	62.9	90.8	84.0	0.81	72.6	89.2	0.86	77.7	90.3	98.6
	Static	0.34	29.4	86.1	47.4	0.71	55.4	78.4	0.99	72.2	72.5	82.6
	StaticUI	0.40	24.7	62.2	44.3	0.69	40.4	58.6	0.96	53.0	55.4	60.5
	Dynamic	0.61	58.1	95.4	76.7	0.77	70.3	91.0	0.85	76.0	89.4	92.2
	DynamicUI	0.61	60.9	100.5	77.7	0.80	75.5	94.9	0.83	78.6	94.4	93.6
	Non-labled StaticUI	0.40	36.4	90.4	53.7	0.80	64.4	80.7	1.11	82.7	74.6	87.4
0.7	Non-hier	0.71	64.1	90.5	83.1	0.85	75.0	88.8	0.87	78.1	89.6	97.3
	Static	0.41	30.0	73.5	48.3	0.70	48.7	69.4	0.98	63.6	64.6	71.7
	StaticUI	0.36	20.9	57.7	37.4	0.70	37.6	54.1	0.99	50.6	51.1	57.3
	Dynamic	0.58	55.8	96.6	73.9	0.73	67.8	92.4	0.83	74.6	89.8	92.5
	DynamicUI	0.60	59.7	98.8	76.1	0.80	74.4	93.2	0.81	75.9	93.9	93.8
	Non-labled StaticUI	0.43	30.8	70.9	51.0	0.73	48.1	66.3	0.97	60.5	62.5	67.3
1.0	Non-hier	0.85	82.8	97.8	92.9	0.87	84.8	97.5	0.88	85.6	97.3	100.2
	Static	0.46	37.5	81.6	60.3	0.71	55.0	77.4	0.88	65.2	74.4	79.8
	StaticUI	0.21	22.4	105.6	24.9	0.86	71.0	83.0	1.50	103.1	68.7	89.0
	Dynamic	0.65	65.6	101.4	81.7	0.77	75.1	98.1	0.82	79.6	96.9	98.5
	DynamicUI	0.75	74.3	99.3	83.5	0.88	83.9	95.4	0.88	84.2	96.0	94.3
	Non-labled StaticUI	0.49	45.5	93.7	67.1	0.77	67.4	87.3	0.98	80.6	82.5	87.4

5.4. Detailed Results on the Distracting Tests

In addition to DAS ratio, Table 4 shows the average AS of distracting utterances and of original utterances in *History*. Table 5 and Table 6 additionally show the AS of the first or last utterances in *History*. Note again that an attention score of 100% for a utterance indicates that this utterance receives an average attention score, e.g. for a dialogue containing 10 utterances, an attention score of 100% indicates that the utterance receives 10% attention out of all.

From Table 4 it is clear that the average AS of the original utterances in *History* varies by model variants. A higher average AS for *History* indicates a lower AS for *Query*. Some models distribute most of the attention to *Query* while some models distribute the attention evenly to both *History* and Query. Normally, *Query* contains more relevant information, so we expect a lower average AS for *History*; however, the average AS for *History* is not the lower the better, since there are still some utterances in *History* that are important for the context. A lower average AS for *History* comes together with a lower average AS for distracting utterances (or a lower DAS), so DAS ratio is better suited for evaluating a model's capability on context attention distribution, since it takes the average AS for original utterances in *History* into account. In Table 4, the models with the lowest DAS ratio also have the lowest average AS for distracting utterances and original utterances, while in Table 5 and Table 6, it is not always the case.

In Table 5 and Table 6, for the distracting test sets where distracting utterances are put in the beginning/end of the context, we show AS for the first/last utterance in *History* to have a clearer comparison. We can see in columns of Frequent: Begin and Rare: Begin that the distracting utterances usually receive lower attention than the first utterance in *History*, while the other original utterances in *History* receive more attention than the first utterance. This indicates a good performance of the model variants. Utterances far away from *Query* are normally distributed lower attention, so in a normal case, it is natural that the utterances that come after the first utterance receive more attention; however, these distracting utterances receive less attention, regardless of the fact that they are placed after the first utterances. It can thus be inferred that most model variants can distinguish distracting utterances as unimportant and distribute less attention to them. Similarly, the last utterances in *History* usually get more attention, while as the columns of Frequent: End and Rare: End show, distracting utterances receive less attention compared to other original utterances in *History*, regardless of that the distracting utterances are placed closer to *Query*.

5.5. Summary of Results

DAS ratio can distinguish conversational agents with similar perplexity on their ability of context attention distribution. In general, models trained with our proposed optimization strategy focus less on distracting utterances and more on original utterances in *History*. For most models, DAS ratios decrease by about 10% when trained with our proposed strategy with a 0.5 or 0.7 probability level. 0.7 is generally the best option for a training inserting probability.

6. CONCLUSIONS AND FUTURE WORKS

We have studied context attention distribution, an essential component of multi-turn modelling for open-domain conversational agents. We have proposed an evaluation metric for context attention distribution based on the distracting test: DAS ratio. We have also improved the performance of context attention distribution for common multi-turn conversational agents through an optimization strategy via reducing the attention loss of self-contained distracting utterances. Extensive experiments show that our proposed strategy achieves improvements on

most models, especially with a training inserting probability level of 0.7. Future works can focus on adapting the proposed evaluation metric and optimization strategy to transformer-based conversational agents.

REFERENCES

[1] Y. Zhang, S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan, "DIALOGPT : Large-scale generative pre-training for conversational response generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, Jul. 2020, pp. 270–278. [Online]. Available: https://www.aclweb.org/anthology/2020.acl-demos.30

[2] X. Zhao, W. Wu, C. Xu, C. Tao, D. Zhao, and R. Yan, "Knowledge-grounded dialogue generation with pre-trained language models," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 3377–3390. [Online]. Available: https://aclanthology.org/2020.emnlp-main.272

[3] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11957

[4] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio, "A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14567

[5] Z. Li, J. Zhang, Z. Fei, Y. Feng, and J. Zhou, "Conversations are not flat: Modeling the dynamic information flow across dialogue utterances," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 128–138. [Online]. Available: https://aclanthology.org/2021.acl-long.11

[6] R. Lowe, N. Pow, I. Serban, and J. Pineau, "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems," in *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, 2015, pp. 285–294. [Online]. Available: http://aclweb.org/anthology/W15-4640

[7] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, Jul. 2002, pp. 311–318. [Online]. Available: https://aclanthology.org/P02-1040

[8] C.-W. Liu, R. Lowe, I. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation," in *Proceedings of the 2016 Conference on EmpiricalMethods in Natural Language Processing*. Association for Computational Linguistics, 2016, pp. 2122–2132. [Online]. Available: http://aclweb.org/anthology/D16-1230

[9] E. Bruni and R. Fernandez, "Adversarial evaluation for open-domain dialogue generation," in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*. Association for Computational Linguistics, 2017, pp. 284–288. [Online]. Available: http://aclweb.org/anthology/W17-5534

[10] R. Lowe, M. Noseworthy, I. V. Serban, N. Angelard-Gontier, Y. Bengio, and J. Pineau, "Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Association for Computational Linguistics, 2017, pp. 1116–1126. [Online]. Available: http://aclweb.org/anthology/P17-1103

[11] Y. Zemlyanskiy and F. Sha, "Aiming to Know You Better Perhaps Makes Me a More Engaging Dialogue Partner," in *Proceedings of the 22nd Conference on Computational Natural Language Learning*. Association for Computational Linguistics, 2018, pp.551–561. [Online]. Available: http://aclweb.org/anthology/K18-1053

[12] Z. Tian, R. Yan, L. Mou, Y. Song, Y. Feng, and D. Zhao, "How to Make Context More Useful? An Empirical Study on Context-Aware Neural Conversational Models," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2017, pp. 231–236. [Online]. Available: http://aclweb.org/anthology/P17-2036

[13] W. Zhang, Y. Cui, Y. Wang, Q. Zhu, L. Li, L. Zhou, and T. Liu, "Context- Sensitive Generation of Open-Domain Conversational Responses," in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2018, pp. 2437–2447. [Online]. Available: http://aclweb.org/anthology/C18-1206

[14] X. Gu, K. M. Yoo, and J.-W. Ha, "DialogBERT: Discourse-aware response generation via learning to recover and rank utterances," in *In Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI 2021)*, 2021.

[15] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112. [Online]. Available: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[18] J. Li, M. Galley, C. Brockett, G. Spithourakis, J. Gao, and B. Dolan, "A Persona-Based Neural Conversation Model," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016, pp. 994–1003. [Online]. Available: http://aclweb.org/anthology/P16-1094

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[20] M. Själander, M. Jahre, G. Tufte, and N. Reissmann, "EPIC: An energy-efficient, highperformance GPGPU computing research infrastructure," 2019.