Learning a Simple and Effective Model for Multi-turn Response Generation with Auxiliary Tasks

Yufan Zhao[1], Can Xu[1*], Wei Wu[2]
[1]Microsoft Corporation, Beijing, China
[2]Meituan, Beijing, China
{yufzhao,caxu}@microsoft.com
wuwei19850318@gmail.com

Abstract

We study multi-turn response generation for open-domain dialogues. The existing state-of-the-art addresses the problem with deep neural architectures. While these models improved response quality, their complexity also hinders the application of the models in real systems. In this work, we pursue a model that has a simple structure yet can effectively leverage conversation contexts for response generation. To this end, we propose four auxiliary tasks including word order recovery, utterance order recovery, masked word recovery, and masked utterance recovery, and optimize the objectives of these tasks together with maximizing the likelihood of generation. By this means, the auxiliary tasks that relate to context understanding can guide the learning of the generation model to achieve a better local optimum. Empirical studies with three benchmarks indicate that our model can significantly outperform state-of-the-art generation models in terms of response quality on both automatic evaluation and human judgment, and at the same time enjoys a much faster decoding process.

1 Introduction

As an important topic in conversational AI, open-domain human-machine conversation is gaining increasing attention from both academia and industry. A common approach to building such a system is to learn a response generation model within an encoder-decoder framework using neural sequence architectures (Sutskever et al., 2014; Vaswani et al., 2017). While the encoder-decoder framework has been successfully applied in various text generation tasks such as machine translation (Vaswani et al., 2017), summarization (Rush et al., 2015), paraphrase generation (Dong et al., 2017), etc., it has to deal with a unique challenge

in the task of response generation: modeling conversation contexts. A conversation context often exhibits a hierarchical structure with dependency existing on both a word-level and an utterance-level. Moreover, as indicated in (Xing et al., 2018; Zhang et al., 2019), information in a context is rather redundant for responding: commonly only a few words and utterances are useful for response generation, and the positions of the relevant words and utterances vary from case to case. To model the hierarchy of conversation contexts, hierarchical recurrent encoder-decoder (HRED) (Serban et al., 2016) extends the vanilla sequence-to-sequence model by a word-level encoder and an utterance-level encoder. Later on, a hierarchical recurrent attention network (HRAN) (Xing et al., 2018) harnesses the decoder of the HRED model with word-level attention and utterance-level attention to dynamically highlight the effect of relevant words and utterances in response synthesis. Very recently, ReCoSa (Zhang et al., 2019) further exploits multi-layer multi-head self-attention[1] to model long-term dependency among utterances and responses. From HRED to HRAN, and then to ReCoSa, the performance of the models in terms of response quality becomes better and better (Zhang et al., 2019), but the models also grow to be more and more complicated. For example, the number of parameters in ReCoSa is more than twice as that in HRED. Thus, when we enjoy the improved performance from the increased complexity, the complexity may also impede the application of the models in some scenarios (e.g., in a mobile scenario).

In this work, we study multi-turn response generation and target on a model that has a simple structure yet can make use of conversation contexts

---

*Corresponding author: Can Xu (caxu@microsoft.com).

[1]The fact that both the encoder and the decoder of ReCoSa contain multiple layers is not highlighted in the paper, but is revealed by the source code released by the authors at https://github.com/zhanghainan/ReCoSa.

as well as the existing deep models. The key idea is to transfer the burden of context understanding from modeling to learning by designing several auxiliary tasks, and leverage the auxiliary tasks as regularization in model estimation. Specifically, the model we use for response generation concatenates utterances in a conversation context as a long sequence, and only exploits one-layer self-attention in encoding and one-layer context attention in decoding. In such a frugal setting, the representation capability of the model shrinks a lot compared with deep transformers. As a remedy, we augment the maximum likelihood estimation (MLE) in learning with objectives from four auxiliary tasks including word order recovery, utterance order recovery, masked word recovery, and masked utterance recovery. In the first two tasks, we predict the correct order of words and utterances from a random shuffle of words in an utterance and a random shuffle of utterances in a context respectively. The goal of the two tasks is to enhance understanding of the sequential dependency among words and utterances within a context. The other two tasks are inspired by the recent breakthrough from BERT (Devlin et al., 2019), in which we randomly mask a word in an utterance and an utterance in a context respectively, and predict the masked word and the masked utterance using the remaining words and utterances. The two tasks may encourage the learning process to pay more attention to semantics of words and utterances in their contexts, and help the learning process find better representations of words and utterances for the generation model. The auxiliary tasks and the MLE task share the encoder of the generation model. Through learning with multiple tasks, optimization for response generation and optimization for context understanding are performed in a joint form. The context understanding related tasks can guide the MLE to achieve a better local optimum, and thus realize superior performance in response generation with a simple neural structure.

We test the proposed approach with three benchmarks including the Ubuntu Dialogue Corpus (Lowe et al., 2015), DailyDialog (Li et al., 2017), and PERSONA-CHAT (Zhang et al., 2018). Evaluation results on all three datasets indicate that our model can significantly outperform state-of-the-art generation models in terms of both automatic evaluation and human judgment. Moreover, with a parameter set even smaller than HRED, our model is 2x faster than ReCoSa in response decoding.

Our contributions in the paper are three-fold: (1) proposal of balancing model complexity and model capability in multi-turn response generation; (2) proposal of four auxiliary learning tasks that transfer context understanding from modeling to learning; and (3) empirical verification of the effectiveness and the efficiency of the proposed model on three benchmarks.

2 Related Work

End-to-end open-domain dialogue generation is built upon the encoder-decoder architecture (Shang et al., 2015; Vinyals and Le, 2015), and the vanilla sequence-to-sequence structure has been widely extended to address challenges such as generic responses (Li et al., 2015; Xing et al., 2017), context modeling (Serban et al., 2016, 2017; Xing et al., 2018; Zhang et al., 2019), and grounding by persona/emotion/knowledge (Li et al., 2016; Zhang et al., 2018; Zhou et al., 2018; Dinan et al., 2018). In this work, we study how to leverage conversation context for multi-turn response generation, which represents a fundamental problem in dialogue generation. Different from the existing work that enhances the representation capability of models through neural architecture engineering, we turn to an orthogonal direction that we keep the generation model simple, and optimize the simple structure by learning with auxiliary tasks that encode context understanding. As a result, our model can provide high-quality responses at a low cost. Before us, there have been a few studies on learning a primary task with auxiliary ones (Rei and Yannakoudakis, 2017; Yu and Jiang, 2016; Ding et al., 2017; Trinh et al., 2018; Mehri et al., 2019; Wu et al., 2019). The work is unique in that through extensive empirical studies, we verified that a simple structure learned with auxiliary tasks can work as well as deep architectures in dialogue generation.

3 Approach

We first formalize the problem in question, and then detail the model and the learning tasks.

3.1 Problem Formalization

Suppose that we have a dataset $\mathcal{D} = \{(\mathcal{U}_i, R_i)\}_{i=1}^N$, where $\mathcal{U}_i = (U_{i,1}, \ldots, U_{i,n})$ denotes a context with $U_{i,j}$ the j-th utterance, and R_i is a response regarding to $\mathcal{U}_i$. The goal is to estimate a generation probability distribution $P(R|\mathcal{U})$ from $\mathcal{D}$, and thus, given a new context $\mathcal{U}$, one can generate a response

for $\mathcal{U}$ following $P(R|\mathcal{U})$. A common practice is to learn $P(R|\mathcal{U})$ by maximizing the log-likelihood of $\mathcal{D}$ (i.e. MLE) which can be formulated as

$$\sum_{i=1}^{N} \log P(R_i|\mathcal{U}_i). \quad (1)$$

When $P(R|\mathcal{U})$ is in a simple structure, only learning with MLE could be insufficient to obtain a model that can well capture the syntax and the semantics of contexts. An evidence is that simple architectures like HRED is much worse than complicated architectures like ReCoSa in terms of response quality, as reported by the existing work (Zhang et al., 2019). Since a simple structure is still favored, we consider aiding the objective given by Equation (1) with extra ones that can reinforce context understanding in the learning process.

3.2 Generation Model

Figure 1 illustrates the architecture of the generation model. In a nutshell, the model is in a transformer-based structure (Vaswani et al., 2017) with one attentive layer (in the transformer layer) in the encoder and one attentive layer in the decoder. The auxiliary tasks, which will be presented later, share the encoder with the generation model. We prefer a transformer-based structure instead of a recurrent structure, because the former is easier to parallelize than the latter, and thus can further enhance efficiency of the model in an online system.

Encoder: we unfold all words in $(\mathcal{U}, R)$ into $\mathcal{W} = (w_1, \ldots, w_m, w_{m+1}, \ldots, w_{m+t})$, where m is the number of words in context $\mathcal{U}$, and t is the number of words in response R. $\forall i \in \{1, \ldots, m+t\}$, w_i is represented by a summation of word embedding, position embedding, and segment embedding:

$$B(w_i) = WE(w_i) + PE(w_i) + SE(w_i), \quad (2)$$

where $WE(w_i)$ represents the word embedding of w_i initialized using GloVe (Pennington et al., 2014), $PE(w_i)$ is the position embedding of w_i which is defined by $Pe(w_i)$, where $e(w_i)$ is a one-hot vector with the only non-zero entry indicating the position of w_i in $\mathcal{W}$, and $P \in \mathbb{R}^{d \times M_p}$ is a randomly initialized matrix with M_p an upper bound of the number of words in a dialogue. $SE(w_i)$ is the segment embedding of w_i defined similarly with the one-hot vector indicating the position of the utterance that contains w_i. The embedding matrix

is then fed to a transformer layer, which can be formulated as

$$
\begin{aligned}
I &= [B(w_1), B(w_2), \ldots, B(w_{m+t})], \\
E &= \text{FNN}(\text{MultiHead}(I, I, I)),
\end{aligned} \quad (3)
$$

where $\text{FNN}(\cdot)$ is a feed-forward neural network and $\text{MultiHead}(Q, K, V)$ is a multi-head attention function with Q a query, K a key, and V a value. To control the receptive field of self-attention in different tasks, we add a mask matrix $M \in \mathbb{R}^{(m+t) \times (m+t)}$ (Dong et al., 2019) in attention computation, and let M determine whether a pair of words can attend to each other according to the learning tasks. Thus, $\text{MultiHead}(Q, K, V)$ is defined by

$$
\begin{aligned}
\text{MultiHead}(Q, K, V) &= \oplus_{i=1}^{K} \text{Head}_i(Q, K, V), \\
\text{Head}_i(Q, K, V) &= \text{Attention}(W_i Q, W_i K, W_i V), \\
\text{Attention}(Q, K, V) &= \text{softmax}(\frac{QK^\top}{\sqrt{d_k}} + M)V,
\end{aligned}
$$
$$(4)$$

where $\oplus$ refers to a concatenation operation, and M is given by

$$
M_{ij} = \begin{cases} 0, & \text{allow to attend,} \\ -\infty, & \text{prevent from attending.} \end{cases} \quad (5)
$$

Decoder: suppose that $(w_{m+1}, \ldots, w_{m+l-1})$ are words generated until step $l-1$, then the next word w_{m+l} is predicted according to:

$$P(w_{m+l}|w_1, \ldots, w_{m+l-1}) = \text{softmax}(W_s O(w_{m+l-1})), \quad (6)$$

where $O(w_{m+l-1})$ is defined by $\text{FNN}(\text{MultiHead}(E(w_{m+l-1}), E, E)$ with $E = [E(w_1), \ldots, E(w_{m+l-1})]$ the output of the encoder, and W_s is a trainable parameter.

3.3 Auxiliary Tasks

Heading for learning the simple structure that can effectively make use of contexts for response generation, we design two kinds of auxiliary tasks including order recovery and masked content recovery. The order recovery tasks aim to enhance the capability of the self-attention module on capturing the sequential relationship among words and utterances, while the masked content recovery tasks can optimize the self-attention module to enhance semantic connection among words and utterances.
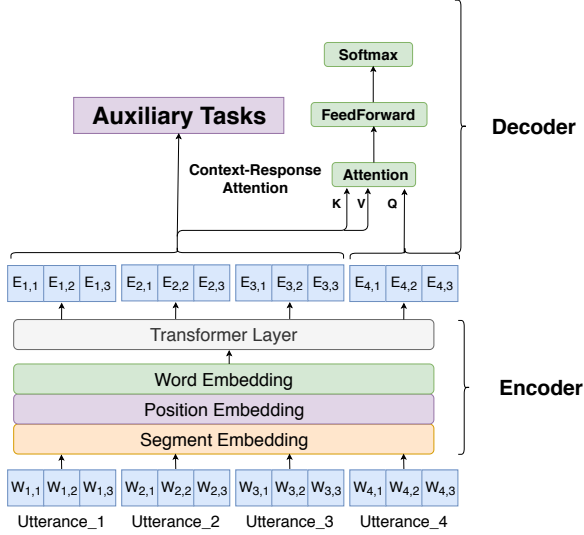
Figure 1: Architecture of the generation model.

Order recovery: a recent study (Sankar et al., 2019) indicates that transformer-based models are insensitive to ordering of words and utterances, which means that the information they learn could be just bag-of-words representations. Thus, we consider recovering the correct order from random shuffling on both a word level and an utterance level to force self-attention to be aware of relative positions of words and utterances in the context.

Word order recovery: Figure 2 (a) illustrates the task. Given a randomly sampled utterance $U = (w_1, \ldots, w_k)$ from a context $\mathcal{U}$, we randomly shuffle the words in U and obtain a disordered utterance $\bar{U} = (\bar{w}_1, \ldots, \bar{w}_k)$. Then, we replace U in $\mathcal{U}$ with $\bar{U}$ and form a corrupt context $\bar{\mathcal{U}}$. The goal of the task is to predict U from $\bar{U}$. The loss of the task can be formulated as

$$\mathcal{L}_{\text{wor}} = -\frac{1}{k} \sum_{i=1}^{k} \log(p(w_i | \bar{U})), \tag{7}$$
$$p(w_i | \bar{U}) = \text{softmax}(W_s E(\bar{w}_i)),$$

where $E(\bar{w}_i)$ is obtained from $E(\bar{\mathcal{U}})$ which is the representation of $\bar{\mathcal{U}}$ given by the encoder of the generation model, W_s is shared with Equation (6).

For this task, the mask matrix M in Equation (4) is defined by:

$$M_{ij} = \begin{cases} 0, & w_i \text{ and } w_j \text{ are in the same utterance,} \\ -\infty, & w_i \text{ and } w_j \text{ are in different utterances.} \end{cases} \tag{8}$$

Utterance order recovery: Figure 2 (d) illustrates the task. Given context $\mathcal{U} = (U_1, \ldots, U_n)$, we randomly shuffle the utterances and obtain a disordered context $\bar{\mathcal{U}} = (U_{o_1}, \ldots, U_{o_n})$. The goal is

to predict the correct positions for utterances in $\bar{\mathcal{U}}$. The prediction model falls in a read-process-write framework (Vinyals et al., 2015). In the reading module, the model first represents $\bar{\mathcal{U}}$ as $\bar{E} = (\bar{E}(w_{1,1}), \ldots, \bar{E}(w_{n,m}))$ via the encoder of the generation model, where $w_{i,j}$ is the j-th word in utterance U_{o_i} (words within an utterance are ordered), and then obtains the representation of utterance U_{o_i} through

$$S_i = \sum_{j=1}^{k_i} \bar{E}(w_{i,j}), \tag{9}$$

where k_i is the number of words in U_{o_i}. $\mathcal{S} = \{S_i\}_{i=1}^{n}$ forms a sentence memory that is accessible by the processing module. The processing module exploits multi-head self-attention and GRU to guarantee the property that vectors retrieved from memory $\mathcal{S}$ will not change if the memory is randomly shuffled. Formally, the processing module is defined by

$$\begin{aligned} \{A_i\}_{i=1}^{n} &= \text{MultiHead}(\mathcal{S}, \mathcal{S}, \mathcal{S}), \\ h_t &= \text{GRU}(h_{t-1}, A_t), \end{aligned} \tag{10}$$

where the last hidden state h_n is permutation invariant regarding to input. The writing module is another GRU that decodes $\{o_1, \ldots, o_n\}$ one by one. At step i, the hidden state $\bar{h}_i$ is defined by

$$\bar{h}_i = \text{GRU}(\bar{h}_{i-1}, [c_i \oplus x_i]), \tag{11}$$

where $\bar{h}_{i-1}$ is the hidden state at step $i-1$ with $\bar{h}_0 = h_n$, x_i is the embedding of o_{i-1} (i.e., the embedding of the ground-truth position of $U_{o_{i-1}}$ in $\mathcal{U}$), and c_i is a context vector which is defined via attention over $\{h_t\}_{t=1}^{n}$:

$$\begin{aligned} c_i &= \sum_{t=1}^{n} a_{i,t} h_t, \\ \{a_{i,t}\}_{t=1}^{n} &= \text{softmax}(\{e_{i,t}\}_{t=1}^{n}), \\ e_{i,t} &= V^{\top} \tanh(W_1 \bar{h}_{i-1} + W_2 h_t + b_1), \end{aligned} \tag{12}$$

where V, W_1, W_2, and b_1 are parameters. The prediction model is finally formulated as

$$\begin{aligned} P(o_i | \{o_1, \ldots o_{i-1}\}, \bar{\mathcal{U}}) &= \text{softmax}(u_i), \\ u_i &= \text{FNN}(\bar{h}_i \oplus x_i \oplus c_i). \end{aligned} \tag{13}$$

The loss function of the task is defined by

$$\mathcal{L}_{\text{uor}} = -\frac{1}{n} \sum_{i=1}^{n} \log(p(o_i | \{o_1, \ldots, o_{i-1}\}, \bar{\mathcal{U}})). \tag{14}$$
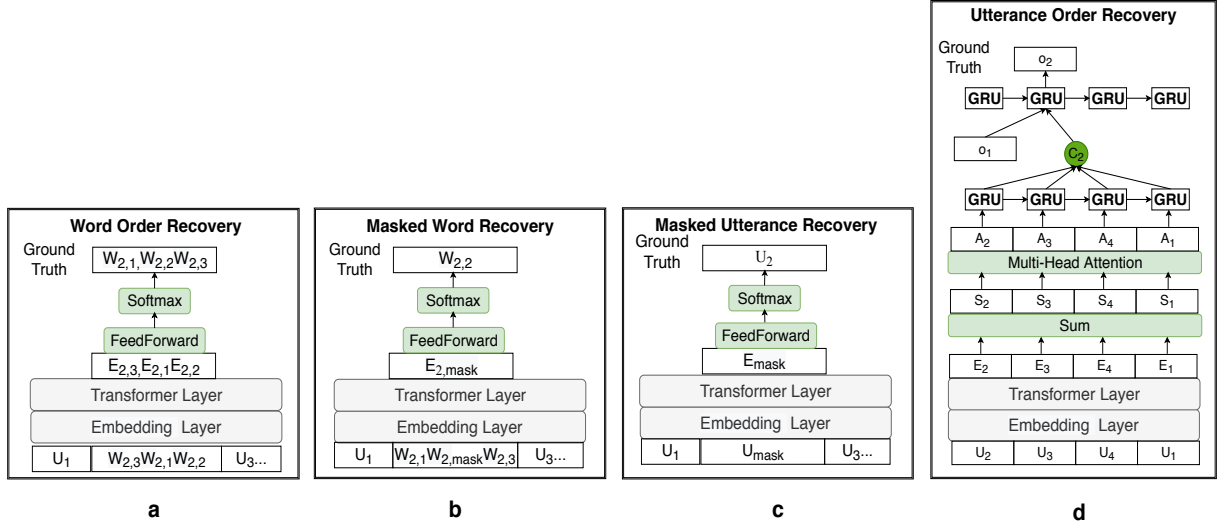
3475

Figure 2: Auxiliary tasks.

For this task and the following ones, M in Equation (4) is defined as a zero matrix meaning that every pair of words can attend to each other in the context.

Masked content recovery: a major challenge in context understanding is the information omission problem (e.g., coreferences) that widely exists in utterances (Su et al., 2019). The challenge requires a model to connect semantically related words and utterances. Thus, we design masked content recovery tasks on both a word level and an utterance level to enhance the self-attention module in terms of awareness of the semantic connections.

- **Word level**: for each utterance in a context, we randomly replace 15% words with a special token [MASK].

- **Utterance level**: we randomly pick an utterance from a context, and replace all words in the utterance with a special token [MASK].

Figure 2 (b) and Figure 2 (c) illustrate the task of masked word recovery (mwr) and the task of masked utterance recovery (mur) respectively. Since the only difference of the two tasks is the input, we present them in a uniform way. Given a context $\mathcal{U} = (w_1, \ldots, w_m)$, suppose that the masked context is $\bar{\mathcal{U}} = (w_1^*, \ldots, w_m^*)$, where $w_i^* = $ [MASK] if w_i is masked, otherwise $w_i^* =$

w_i, then, the loss of the tasks can be formulated as

$$
\begin{aligned}
\mathcal{L}_x &= -\frac{1}{k} \sum_{i=1}^{m} \mathbb{I}[w_i^*=\text{[MASK]}] \log(p(w_i|\bar{\mathcal{U}})), \\
k &= \sum_{i=1}^{m} \mathbb{I}[w_i^*=\text{[MASK]}], \\
p(w_i|\bar{\mathcal{U}}) &= \text{softmax}(W_s E(w_i^*)),
\end{aligned}
\tag{15}
$$

where $E(w_i^*)$ is the representation of w_i^* obtained by passing $\mathcal{U}$ through the encoder of the generation model, $x \in \{\text{mwr}, \text{mur}\}$ indexes the two tasks, $\mathbb{I}[\cdot]$ is an indicator function, and W_s is shared with Equation (6).

3.4 Learning Objective

The full loss function is finally defined by:

$$
\begin{aligned}
\mathcal{L}_{\text{full}} &= \text{MLE} + \alpha \mathcal{L}_{\text{aux}}, \\
\mathcal{L}_{\text{aux}} &= \mathcal{L}_{\text{wor}} + \mathcal{L}_{\text{uor}} + \mathcal{L}_{\text{mwr}} + \mathcal{L}_{\text{mur}},
\end{aligned}
\tag{16}
$$

where α is a hyper-parameter as a trade-off between MLE and the objectives of the auxiliary tasks. The learning algorithm is summarized in Algorithm 1, where Θ refers to a set of parameters including both the parameters of the generation model and the parameters of the auxiliary objectives.

4 Experiments

We conduct experiments on DailyDialog (Li et al., 2017), PERSONA-CHAT (Zhang et al., 2018), and the Ubuntu Dialogue Corpus (UDC) (Lowe et al., 2015), and compare our model with state-of-the-art baselines in terms of response quality, parameter size, and decoding speed.

3476

Algorithm 1: Optimization Algorithm

Input: Training data $\mathcal{D}$, GlobalMaxStep T_1, AuxTrainEpoch T_2, InitialRate α, BatchNumPerEpoch N

Init: Θ

1   $t = 0$
2   $\alpha = 1.0$
3   $d = \alpha/(T_2 * N)$
4   **while** $t < T_1$ **do**
5   Randomly sample a mini-batch k from $\mathcal{D}$.
6   **if** $\alpha > 0$ **then**
7 | Compute $\mathcal{L}_{\text{aux}}$.
8   Compute MLE.
9   Update the parameters of the model with respect to $\mathcal{L}_{\text{full}}$ using Adagrad.
10     $\alpha = max(0, \alpha - d)$
11     $t = t + 1$

Output: Θ

	DailyDialog	PERSONA-CHAT	Ubuntu
# dialogues for training	44,050	95,682	3980,000
# dialogues for validation	4,176	11,602	10,000
# dialogues for test	3,864	11,152	10,000
avg. # utter. per dialogue	7.0	9.4	4.3
avg. utter. length	13.6	14.5	16.6

Table 1: Statistics of the datasets.

4.1 Datasets

Both DailyDialog and PERSONA-CHAT are open domain datasets. Dialogues in DailyDialog cover a wide range of topics in daily scenarios and resemble human communications in their daily life; while PERSONA-CHAT contains multi-turn chit-chat conversations between turkers according to their assigned profiles. Since the focus of the work is how to leverage conversation history for response generation, we just append the profiles (the original ones) to the corresponding dialogues as an extension of contexts. To control the length of the dialogues and increase the number of instances, we slide a window on the training/validation/test dialogues in both datasets, and split a dialogue longer than 11 utterances to multiple instances (i.e., the window size is 11). Moreover, we also truncate long utterances with the first 25 words kept. Vocabularies are formed with all words appearing in the entire data and are shared by contexts and responses. The vocabulary size of DailyDialog is $25,000$ and the vocabulary size of PERSONA-CHAT is $18,750$. The UDC data are collected from Ubuntu chat logs with two-person multi-turn conversations about Ubuntu-related problems. Here we use the same data as in (Zhang et al., 2019). Table 1 reports some statistics of the three datasets.

4.2 Baselines

We select several multi-turn response generation models as baselines: (1) **HRED**[2]: hierarchical encoder-decoder proposed in (Serban et al., 2016); (2) **VHRED**[3]: an extension of HRED that factorizes response generation with latent variables (Serban et al., 2017); (3) **HRAN**[4]: hierarchical encoder-decoder equipped with a hierarchical attention mechanism (Xing et al., 2018); (4) **ReCoSa**[5]: a hierarchical transformer-based model that exhibits state-of-the-art performance on benchmarks (Zhang et al., 2019); and (5) **SSN**: a very recent study on enhancing dialogue generation learning with self-supervision signals extracted from utterance order (Wu et al., 2019).

4.3 Implementation Details

We train the baselines and our model on RTX 2080, and initialize word embedding with GloVe vectors (Pennington et al., 2014). In our model, the dimension of all vectors is set as 512. The number of heads in multi-head attention is set as 8. We adopt the Adagrad algorithm (Duchi et al., 2011) in optimization with a learning rate 0.05 and a batch size 80/60/32 in DailyDialog/PERSONA-CHAT/Ubuntu. All models are tuned on the validation sets according to perplexity. We stop training if the perplexity does not drop in three consecutive epochs. The GlobalMaxStep T_1 is set as 50k. The AuxTrainEpoch T_2 is set as 30. The BatchNumPerEpoch N is $551/1595/124,375$ for DailyDialog/PERSONA-CHAT/Ubuntu.

4.4 Evaluation Metrics

We evaluate the performance of the models in terms of response quality with both automatic metrics and human judgment. In automatic evaluation, besides BLEU-4 (Papineni et al., 2002) and perplexity (Sutskever et al., 2014), we follow (Serban et al.,

---

[2] https://github.com/hsgodhia/hred
[3] https://github.com/julianser/hed-dlg-truncated
[4] https://github.com/LynetteXing1991/HRAN
[5] https://github.com/zhanghainan/ReCoSa

Dataset	Model	PPL	BLEU	Distinct-1	Distinct-2	Average	Greedy	Extrema	Parameter size	Decoding speed
DailyDialog	HRED	56.22	0.535	1.553	3.569	81.393	65.546	48.109	34.5M	14.79ms
	HRAN	47.23	0.447	1.953	7.400	83.460	67.239	**49.599**	38.2M	17.15ms
	VHRED	44.79	0.997	1.299	6.113	83.866	67.186	48.570	34.8M	15.67ms
	SSN	44.28	1.250	2.309	7.266	72.796	**73.069**	44.260	20.0M	12.69ms
	ReCoSa	42.34	1.121	1.987	10.180	84.763	67.557	48.957	73.8M	40.89ms
	Our Model	**38.60**	**1.658**	**3.457**	**14.954**	**85.224**	69.518	49.069	20.3M/14.4M	12.15ms
PERSON-CHAT	HRED	46.04	1.279	0.164	0.450	83.329	64.486	47.132	28.3M	13.14ms
	HRAN	41.94	1.997	0.235	0.771	82.850	65.556	47.882	33.1M	18.43ms
	VHRED	42.07	2.181	0.312	1.915	82.995	65.578	46.810	28.8M	20.27ms
	SSN	47.90	2.288	0.637	2.623	**85.002**	66.752	47.461	15.2M	15.82ms
	ReCoSa	34.19	2.258	0.915	4.217	83.963	66.498	48.163	68.7M	39.38ms
	Our Model	**33.23**	**2.434**	**1.279**	**5.816**	83.632	66.778	**48.552**	18.4M/12.5M	13.89ms
Ubuntu	HRED	58.23	0.874	0.602	2.724	76.187	62.869	37.508	24.1M	25.09ms
	HRAN	48.14	0.922	0.472	2.217	76.654	62.145	37.282	29.5M	31.07ms
	VHRED	52.34	0.906	0.571	2.933	76.496	63.051	36.039	24.7M	30.47ms
	SSN	57.82	**1.681**	0.557	2.370	76.431	61.597	35.976	12.3M	21.11ms
	ReCoSa	43.67	0.911	0.722	4.439	77.619	**63.239**	36.742	60.6M	45.34ms
	Our Model	**40.94**	1.625	**0.783**	**5.151**	**78.754**	62.738	**38.538**	14.4M/8.5M	22.98ms

Table 2: Evaluation results on automatic metrics. Numbers in bold indicate the best performing model on the corresponding metrics.

2017) and employ Embedding Average (Average), Embedding Extrema (Extrema), and Embedding Greedy (Greedy) as metrics. We also follow (Li et al., 2015) and measure the informativeness of responses with distinct-1 and distinct-2 that are calculated as the ratios of distinct unigrams and bigrams. In human evaluation, we randomly sample 500 dialogues from each of the three test sets, and recruit 3 native speakers as human annotators. For each context in the 500 dialogues, each annotator compares a response from our model and a response from a baseline model. The two responses are top one results from greedy search, and are randomly shuffled to hide their sources. The annotators judge which response is better based on informativeness, consistency, and fluency of the responses. If an annotator cannot tell which response is better, he/she is required to label a "tie". Each annotator individually judges 500 pairs for all combinations of our model and baseline models. In total, each one labels 2, 500 pairs for one dataset. Fleiss' kappa (Fleiss and Cohen, 1973) is employed to measure agreement among the annotators.

In addition to response quality, we also compare our model with baselines on decoding speed. We calculate the average prediction time per word in response generation using all dialogues in the test sets. The efficiency comparison is conducted on a GPU environment with a single RTX 2080.

4.5 Evaluation Results

Table 5 reports evaluation results on automatic metrics. Our model outperforms all baseline methods on most of the metrics on all the three datasets.

The last two columns of the tables compare different models in terms of parameter size and decoding speed. Note that in training, the auxiliary tasks contain parameters outside the generation model. Therefore, in the column of parameter size, we report two numbers for our model with the one before "/" parameter size in training and the one after "/" parameter size of the generation model. It is remarkable that the parameter size of our model, even in training, is smaller than HRED. In spite of this, the model still outperforms ReCoSa with only $19.5\%/18.2\%/14.0\%$ parameters on the DailyDialog/PERSONA-CHAT/Ubuntu data. This is because (1) the auxiliary tasks can effectively aid the learning of the generation model in our method; and (2) ReCoSa, although in a deep structure, is still inadequate in terms of context modeling due to the RNN-based encoder and the only utterance-level attention. Besides the superior performance on response quality, our model also enjoys a fast decoding process, thanks to the small model size. In terms of decoding speed, our model is comparable with HRED, and 2x faster than ReCoSa. The generation model of SNN is just a simple RNN sequence-to-sequence with one layer encoder and one layer decoder. Therefore, our model is comparable with SSN in terms of complexity and speed. However, SSN is worse than our model on response quality due to (1) the RNN-based seq2seq model in SSN is worse than a transformer-based structure on the benchmarks used in this work, which has been indicated by Sankar et al. (2019); (2) SSN only considers utterance order, while we also leverage word order, word content, and utterance content in

3478

DailyDialog

model variant	PPL	BLEU	distinct-1	distinct-2	Average	Greedy	Extrema
full tasks	38.60	1.658	3.457	14.954	85.224	69.518	49.069
- masked word recovery	38.37	1.365	2.629	11.135	85.270	69.901	49.495
- masked utterance recovery	39.06	1.407	2.980	12.544	85.143	69.667	49.791
- word order recovery	41.53	1.082	2.769	11.166	85.020	69.417	49.567
- utterance order recovery	38.69	1.215	2.551	9.764	85.253	69.678	49.644
- all tasks	46.58	0.903	1.775	7.136	84.042	69.017	48.467

PERSONA-CHAT

model variant	PPL	BLEU	distinct-1	distinct-2	Average	Greedy	Extrema
full tasks	33.23	2.434	1.279	5.816	83.632	66.778	48.552
- masked word recovery	34.74	2.429	1.018	4.764	82.841	66.177	48.610
- masked utterance recovery	33.49	2.638	1.045	5.412	83.402	66.862	48.810
- word order recovery	35.06	2.355	1.028	4.698	82.503	66.011	48.350
- utterance order recovery	33.24	2.484	1.054	5.011	82.652	66.025	47.927
- all tasks	37.16	1.928	0.938	4.141	82.104	65.899	47.162

Ubuntu

model variant	PPL	BLEU	distinct-1	distinct-2	Average	Greedy	Extrema
full tasks	40.94	1.625	0.783	5.151	78.754	62.738	38.538
- masked word recovery	47.02	1.135	0.404	2.195	74.735	61.683	37.914
- masked utterance recovery	42.48	1.543	0.519	2.419	76.381	62.203	37.482
- word order recovery	48.57	0.962	0.325	1.537	77.615	62.819	38.651
- utterance order recovery	52.04	1.023	0.359	1.609	74.982	59.384	36.825
- all tasks	57.32	0.851	0.391	1.765	73.582	62.581	37.268

Table 3: Results of ablation study.

learning. In fact, we find that the proposed auxiliary tasks can improve a 2-layer (one for encoder and one for decoder) RNN-based seq2seq model as well, as reported in Supplementary Material. On most metrics, RNN with full auxiliary tasks is better than SSN but worse than the proposed model.

Table 4 summarizes human evaluation results. We can see that our model outperforms all baseline models, and most of the kappa values exceed 0.6 indicating substantial agreement among the annotators. Based on the annotation results, we find that our model tends to generate diverse and context consistent responses, indicating the effect of the auxiliary tasks.

4.6 Discussions

To further understand the merit of the auxiliary tasks, we make some analysis regarding to the following questions: **Q1** how do the simple architecture learned with the auxiliary tasks compare with a deep architecture; **Q2** if learning with the auxiliary tasks can also improve deep architectures; and **Q3** how different auxiliary tasks affect the performance of the model.

Answer to Q1: we aim to move one step fur-

ther to understand how the auxiliary tasks enhance the capability of the simple generation model on context understanding. While this is not trivial for neural models, we assume that one can let a transformer-based model capture more semantics in contexts by stacking more layers in the encoder, and examine to what extent the simple model learned with the auxiliary tasks is equivalent to a deep architecture. Figure 3 compares our model with deep architectures in terms of perplexity on the three datasets, in which we get the deep architectures by stacking transformer layers in the encoder of our model. The dotted lines represent our model learned with the auxiliary tasks, and the solid lines represent the deep architectures learned with MLE. Approximately, our model is equivalent to a deep model with a 4-layer encoder on the DailyDialog data, a 6-layer encoder on the PERSONA-CHAT data, and a 7-layer encoder on the Ubuntu data.

Answer to Q2: since the auxiliary tasks are useful for the simple model, it is also interesting to check if they work as well for deep architectures. Figure 3 shows the results, in which the dash-dotted lines represent the deep architectures learned with the full auxiliary tasks. First of all, we can conclude
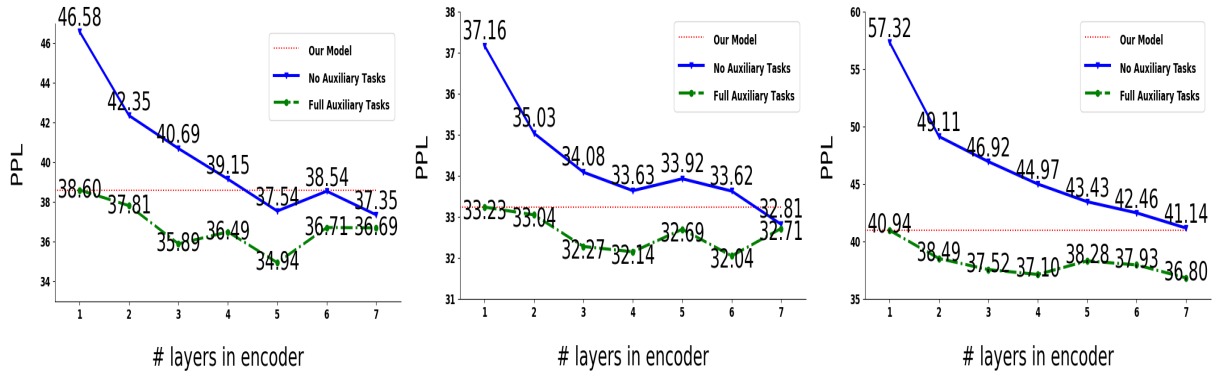
Figure 3: Performance of deep architectures. (a) DailyDialog; (b) PERSONA-CHAT; (c) Ubuntu

DailyDialog				
models	win	loss	tie	kappa
Our Model v.s. HRED	0.42	0.13	0.45	0.675
Our Model v.s. VHRED	0.38	0.19	0.43	0.634
Our Model v.s. HRAN	0.31	0.16	0.53	0.587
Our Model v.s. SSN	0.36	0.22	0.42	0.638
Our Model v.s. ReCoSa	0.34	0.22	0.44	0.733
PERSONA-CHAT				
models	win	loss	tie	kappa
Our Model v.s. HRED	0.45	0.16	0.39	0.867
Our Model v.s. VHRED	0.39	0.21	0.40	0.650
Our Model v.s. HRAN	0.36	0.23	0.41	0.621
Our Model v.s. SSN	0.49	0.12	0.39	0.695
Our Model v.s. ReCoSa	0.39	0.29	0.32	0.566
Ubuntu				
models	win	loss	tie	kappa
Our Model v.s. HRED	0.49	0.14	0.37	0.692
Our Model v.s. VHRED	0.48	0.18	0.34	0.603
Our Model v.s. HRAN	0.47	0.13	0.40	0.612
Our Model v.s. SSN	0.45	0.18	0.37	0.698
Our Model v.s. ReCoSa	0.39	0.27	0.34	0.672

Table 4: Human evaluation results. The ratios are calculated by combining annotations from three judges together.

that the auxiliary tasks are also useful for deep architectures, since there is clear PPL drop for the same models learned with and without (i.e., the solid lines) the auxiliary tasks. Secondly, the auxiliary tasks are more useful for simple structures, since the gap between the same models learned with and without the tasks becomes smaller and smaller when the number of encoding layers increases. The results indicate that after stacking enough layers, the effect of the auxiliary tasks is overwhelmed by the model itself. Therefore, the merit of the auxiliary tasks is to allow us to learn a generation model that enjoys both efficacy and efficiency, which is exactly the goal of the work. Improvement with respect to the number of layers of the encoder on UDC is more steady than that on

DailyDialog and PERSON-CHAT. This is because the training set of UDC is much larger than those of the other two datasets.

Answer to Q3: we keep the architecture of the generation model and remove the objectives of the auxiliary tasks one at a time from the full learning objective given by Equation (16). Table 3 reports the ablation results. First of all, all auxiliary tasks are useful as removing any of them will cause a performance drop. When all auxiliary tasks are removed, the approach degenerates to learning a 2-layer transformer architecture through MLE. Without any optimization on context understanding, the simple structure is worse than ReCoSa. Secondly, on DailyDialog and UDC, order recovery tasks are more crucial than content recovery tasks due to the order insensitive nature of self-attention. Finally, on PERSONA-CHAT, word-level recovery tasks matter more than utterance-level recovery tasks. This might stem from the fact that in PERSONA-CHAT, dialogues highly depend on the profiles used as contexts. In many cases, utterances are just formed by copying a proportion of words from the profiles. Thus, recognizing the semantic connections and the relationship among words in contexts is more critical for the data.

5 Conclusions

We propose a simple generation model with order recovery and masked content recovery as auxiliary tasks. Evaluation results on three benchmarks indicate that our model can significantly outperform state-of-the-art deep generation models in terms of both response quality and decoding speed.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Ying Ding, Jianfei Yu, and Jing Jiang. 2017. Recurrent neural networks with auxiliary labels for cross-domain opinion target extraction. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. Learning to generate product reviews from attributes. In *EACL*, pages 623–632.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. In *NAACL*, pages 110–119.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *ACL*, pages 994–1003.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. In *IJCNLP*, pages 986–995.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. 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*, pages 285–294.

Shikib Mehri, Evgeniia Razumovsakaia, Tiancheng Zhao, and Maxine Eskenazi. 2019. Pretraining methods for dialog context representation learning. *arXiv preprint arXiv:1906.00414*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Marek Rei and Helen Yannakoudakis. 2017. Auxiliary objectives for neural error detection models. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 33–43.

Alexander M Rush, Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389.

Chinnadhurai Sankar, Sandeep Subramanian, Christopher Pal, Sarath Chandar, and Yoshua Bengio. 2019. Do neural dialog systems use the conversation history effectively? an empirical study. *arXiv preprint arXiv:1906.01603*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

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

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *ACL*, pages 1577–1586.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance rewriter. *arXiv preprint arXiv:1906.07004*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*, NIPS'14, pages 3104–3112, Cambridge, MA, USA. MIT Press.

Trieu Trinh, Andrew Dai, Thang Luong, and Quoc Le. 2018. Learning longer-term dependencies in rnns with auxiliary losses. In *International Conference on Machine Learning*, pages 4972–4981.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NIPS*, pages 5998–6008. Curran Associates, Inc.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2015. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*.

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Jiawei Wu, Xin Wang, and William Yang Wang. 2019. Self-supervised dialogue learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3857–3867.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *AAAI*, pages 3351–3357.

Chen Xing, Wei Wu, Yu Wu, Ming Zhou, Yalou Huang, and Wei-Ying Ma. 2018. Hierarchical recurrent attention network for response generation. In *AAAI*, pages 5610–5617.

Jianfei Yu and Jing Jiang. 2016. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 236–246.

Hainan Zhang, Yanyan Lan, Liang Pang, Jiafeng Guo, and Xueqi Cheng. 2019. Recosa: Detecting the relevant contexts with self-attention for multi-turn dialogue generation. In *ACL*, pages 3721–3730.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *ACL*, pages 2204–2213.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *AAAI*, pages 730–738.

A   RNN with Auxiliary Tasks

As a follow-up investigation, we are curious about if the auxiliary tasks can enhance the performance of other simple architectures in the task of multi-turn response generation. Table 5 reports the results on the three benchmarks, where the simple architecture is an RNN-base seq2seq model with one layer encoder and one layer decoder. The architecture of the model is the same as the one in SSN, that is the encoder is defined with a bi-directional GRU, the decoder is defined with a unidirectional GRU, and the decoder is equipped with an attention mechanism on the input context. From the results, we can see that the auxiliary tasks are also useful for the RNN architecture, although it is still worse than the proposed transformer-based architecture under the same learning protocol. On most metrics, the RNN model is better than SSN, since it leverages signals from full auxiliary tasks.

	Model	PPL	BLEU	Distinct-1	Distinct-2	Average	Greedy	Extrema
DailyDialog	RNN	47.69	0.668	1.001	3.563	80.191	71.211	45.526
	RNN+Auxiliary Tasks	42.46	1.271	3.153	12.454	75.259	72.077	45.490
	SSN	44.28	1.250	2.309	7.266	72.796	73.069	44.260
	Our Model	38.60	1.658	3.457	14.954	85.224	69.518	49.069
PERSON-CHAT	RNN	42.51	1.869	0.172	0.501	81.855	64.284	46.504
	RNN+Auxiliary Tasks	38.20	2.356	0.986	4.037	84.907	66.951	48.419
	SSN	47.90	2.288	0.637	2.623	85.002	66.752	47.461
	Our Model	33.23	2.434	1.279	5.816	83.632	66.778	48.552
Ubuntu	RNN	62.56	0.971	0.398	1.218	74.318	59.027	34.331
	RNN+Auxiliary Tasks	55.44	1.479	0.602	3.494	76.494	62.381	38.139
	SSN	57.82	1.681	0.557	2.370	76.431	61.597	35.976
	Our Model	40.94	1.625	0.783	5.151	78.754	62.738	38.538

Table 5: Evaluation results on automatic metrics.