

Response Generation by Context-aware Prototype Editing

Yu Wu^{†◇}, Furu Wei[‡], Shaohan Huang[‡], Zhoujun Li^{†◇}, Ming Zhou[‡]

[†]State Key Lab of Software Development Environment, Beihang University, Beijing, China

[◇] Authors are supported by AdeptMind Scholarship

[‡] Microsoft Research, Beijing, China

{wuyu,lizj}@buaa.edu.cn {fuwei, shaohan, mingzhou}@microsoft.com

Abstract

Open domain response generation has achieved remarkable progress in recent years, but sometimes yields short and uninformative responses. We propose a new paradigm for response generation, that is response generation by editing, which significantly increases the diversity and informativeness of the generation results. Our assumption is that a plausible response can be generated by slightly revising an existing response prototype. The prototype is retrieved from a pre-defined index and provides a good start-point for generation because it is grammatical and informative. We design a response editing model, where an edit vector is formed by considering differences between a prototype context and a current context, and then the edit vector is fed to a decoder to revise the prototype response for the current context. Experiment results on a large scale dataset demonstrate that the response editing model outperforms generative and retrieval-based models on various aspects.

1 Introduction

In recent years, non-task oriented chatbots, focused on responding to humans intelligently on a variety of topics, have drawn much attention from both academia and industry. Existing approaches can be categorized into generation-based methods (Shang et al., 2015; Vinyals and Le, 2015; Serban et al., 2016; Sordoni et al., 2015; Xing et al., 2017; Serban et al., 2017; Xing et al., 2018) which generate a response from scratch, and retrieval-based methods (Hu et al., 2014; Lowe et al., 2015; Yan et al., 2016; Zhou et al., 2016; Wu et al., 2017) which select a response from an existing corpus.

Context	My friends and I went to some vegan place for dessert yesterday.
Prototype context	My friends and I <u>had Tofu and vegetables</u> at a vegan place <u>nearby</u> yesterday.
Prototype response	Raw green vegetables are very beneficial for your health.
Revised response	Desserts are very bad for your health.

Table 1: An example of context-aware prototypes editing. Underlined words mean they do not appear in the original context, while ~~words with strikethrough~~ mean they are not in the prototype context. Words in bold represent they are modified in the revised response.

Since retrieval-based approaches are severely constrained by a pre-defined index, generative approaches become popular in recent years. Traditional generation-based approaches, however, do not easily generate long, diverse and informative responses, which is referred to as “safe response” problem (Li et al., 2016a).

To address this issue, we propose a new paradigm, prototype-then-edit, for response generation. Our motivations are three-folds: 1) We observe that most of responses can be represented as transformations of other responses, so it is unnecessary to generate a response from scratch; 2) Human-written responses, called “prototypes” in this paper, are informative, diverse and grammatical which do not suffer from short and generic issues. Hence, generating responses by editing such prototypes can avoid the “safe response” problem. 3) In order to adapt current context, slight revision should be done on the prototype, in which irrelevant words are removed and appropriated words are inserted.

Inspired by this idea, we formulate the response generation process as follows. Given a conversational context c , we first retrieve a similar context c' and its associated response r' from a pre-defined index, and then invoke an edit vector by analyzing the differences between c and c' . After that, we revise the prototype conditioning on the edit vector. We further illustrate how our idea works with an example in Table 1. It is obvious that the major difference between c and c' is what the speaker eats, so the phrase “raw green vegetables” in r' should be replaced by “desserts” in order to reply current context c . We hope that the decoder language model could remember the collocation of “desserts” and “bad for health”, so as to replace “beneficial” with “bad” in the revised response. This paradigm can be seen as an ensemble of a retrieval based approach and a generation based approach, that not only inherits the fluency and informativeness advantages from retrieval results, but also enjoys the flexibility of generation results.

In practice, there is a problem that must be solved, which is how to revise the prototype in a conditional setting¹. Prior work (Guu et al., 2017) has figured out how to edit prototype in an unconditional setting, which cannot be adopted to the response generation task directly. Our idea is that differences between responses strongly correlates with differences in their contexts, meaning if a word in prototype context is changed, its related words in the response are probably modified in the editing. We realize this idea by designing a context-aware editing model that is built upon a encoder-decoder model augmented with an editing vector. The edit vector is computed by the weighted sum of insertion word embeddings (words in prototype context but not in current context) and deletion word embeddings (words in current context but not in prototype context). Larger weights mean that the editing model should pay more attention on corresponding words in revision. For instance, we wish “dessert”, “Tofu” and “vegetables” get larger weights than “and” in Table 1. The encoder learns the prototype representation with a gated recurrent unit (GRU), and feeds the representation to a decoder together with the edit vector. The decoder is a GRU language

model, that regards the concatenation of last step word embedding and the edit vector as inputs, and predicts the next word with an attention mechanism.

Our experiments are conducted on a large scale Chinese conversation corpus comprised of 10 million context-response pairs. Given a response, we find its neighbors based on lexicon similarities to build the prototype-target parallel data for training. In terms of testing, we retrieve a similar context and adapt its response to current context by editing. The experiments show that our method outperforms traditional generative models on relevance, diversity and originality. We further find that the revised response achieves better relevance compared to its prototype, demonstrating that the editing process is not trivial.

Our contributions are listed as follows: 1) this is the first work to propose a prototype-then-edit paradigm for response generation, that is significantly different from those that generate responses from scratch; 2) we elaborate a simple but effective context-aware editing model for response generation; 3) we empirically verify the effectiveness of our method.

2 Related Work

Research on chatbots goes back to the 1960s when ELIZA was designed (Weizenbaum, 1966) that uses a huge amount of hand-crafted templates and rules. Recently, researchers have paid more and more attention on data-driven approaches (Ritter et al., 2011; Ji et al., 2014) due to their superior scalability. Most of these methods are classified as retrieval-based methods (Ji et al., 2014; Yan et al., 2016) and generation methods (Li et al., 2016b, 2017; Mou et al., 2016; Zhou et al., 2017). The former one aims to select a relevant response using a matching model, while the latter one generates a response with natural language generative models.

Prior works on retrieval-based methods mainly focus on the matching model architecture for single turn conversation (Hu et al., 2014) and multi-turn conversation (Lowe et al., 2015; Zhou et al., 2016; Wu et al., 2017). For the studies of generative methods, a huge amount of work aims to mitigate the “safe response” issue from different perspectives. Most of work build models under a sequence to sequence framework (Sutskever et al., 2014), and introduce other elements, such as latent variables (Serban et al., 2017), topic infor-

¹conditional setting means the editor should consider the context (dialogue history) except in a response itself in the revision. Editor only considers a sentence itself in the unconditional setting

mation (Xing et al., 2017), dynamic vocabulary (Wu et al., 2018) and other contents (Yao et al., 2017; Mou et al., 2016) to increase response diversity. Furthermore, the reranking technique (Li et al., 2016a), reinforcement learning technique (Li et al., 2016b) and adversarial learning technique (Li et al., 2017; Xu et al., 2017) have also been applied to response generation. Apart from work on “safe response”, there is a growing body of literature on automatic response evaluation (Tao et al., 2018; Lowe et al., 2017), style transfer (Fu et al., 2018; Wang et al., 2017) and emotional response generation (Zhou et al., 2017). Retrieval results have been used for dialogue generation in (Song et al., 2016), where the decoder takes representations of retrieval results and the original query as inputs. Compared to our work, it neither edits a prototype nor utilizes the difference between two contexts. In general, previous work generates a response from scratch either left-to-right or conditioned on a latent vector, whereas our approach is the first one to generate a response by editing a prototype, facilitating a new direction for response generation.

Recently, some researches have explored natural language generation by editing (Guu et al., 2017; Su et al., 2018; Liao et al., 2018). A typical approach follows a writing-then-edit paradigm, that utilizes one decoder to generate a draft from scratch and uses another decoder to revise the draft (Xia et al., 2017). The other approach follows a retrieval-then-edit paradigm, that uses a Seq2Seq model to edit a prototype retrieved from a corpus (Guu et al., 2017; Li et al., 2018; Cao et al., 2018). To the best of our knowledge, we are the first to explicitly leverage context differences to edit prototypes, and this approach can be applied to other Seq2Seq tasks, such as summarization, paraphrase generation and machine translation.

3 Background

Before introducing our approach, we first briefly describe state-of-the-art natural language editing method (Guu et al., 2017). Given a sentence pair (x, x') , our goal is to obtain sentence x by editing the prototype x' . The general framework is built upon a Seq2Seq model with an attention mechanism, which takes x' and x as source sequence and target sequence respectively. The main difference is that the generative probability of a vanilla Seq2Seq model is $p(x|x')$ whereas the probabil-

ity of the edit model is $p(x|x', z)$ where z is an edit vector sampled from a pre-defined distribution like variational auto-encoder. In the training phase, the parameter of the distribution is conditional on the context differences. We first define $I = \{w|w \in x \wedge w \notin x'\}$ as an insertion word set, where w is a word added to the prototype, and $D = \{w'|w' \in x' \wedge w' \notin x\}$ is a deletion word set, where w is a word deleted from the prototype. Subsequently, we compute an insertion vector $\vec{i} = \sum_{w \in I} \Psi(w)$ and a deletion vector $\vec{d} = \sum_{w' \in D} \Psi(w')$ by a summation over word embeddings in two corresponding sets, where $\Psi(\cdot)$ transfers a word to its embedding. Then, the edit vector z is sampled from a distribution whose parameters are governed by the concatenation of \vec{i} and \vec{d} . Finally, the edit vector and output of the encoder are fed to the decoder to generate x .

For response generation, which is a conditional setting of text editing, an interesting question raised, that is how to generate the edit by considering contexts. We will introduce our motivation and model in details in the next section.

4 Approach

4.1 Model Overview

Suppose that we have a data set $\mathcal{D} = \{(C_i, R_i)\}_{i=1}^N$. $\forall i$, (C_i, R_i) comprises a context $C_i = (c_{i,1}, \dots, c_{i,l})$ and its response $R_i = (r_{i,1}, \dots, r_{i,l})$, where $c_{i,j}$ is the j -th word of the context C_i and $r_{i,j}$ is the j -th word of the response R_i . It should be noted that C_i can be either a single turn input or a multiple turn input. As the first step, we assume C_i is a single turn input in this work, and leave the verification of the same technology for multi-turn response generation to future work. Our full model is shown in Figure 1, consisting of a prototype selector \mathcal{S} and a context-aware neural editor \mathcal{E} . Given a new conversational context C , we first use \mathcal{S} to retrieve a context-response pair $(C_i, R_i) \in \mathcal{D}$. Then, the editor \mathcal{E} calculates an edit vector $z_i = f(C_i, C)$ to encode the information about the differences between C_i and C . Finally, we generate a response according to the probability of $p(R|z_i, R_i)$. In the following, we will elaborate how to design the selector \mathcal{S} and the editor \mathcal{E} .

4.2 Prototype Selector

A good prototype selector \mathcal{S} plays an important role in the prototype-then-edit paradigm. We use

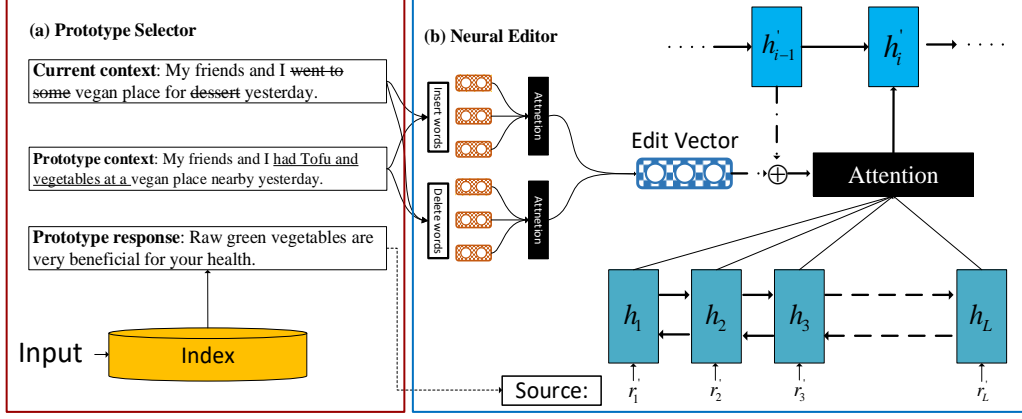


Figure 1: Architecture of our model.

different strategies to select prototypes for training and testing. In testing, as we described above, we retrieve a context-response pair (C', R') from a pre-defined index for context C according to the similarity of C and C' . Here, we employ Lucene² to construct the index and use its inline algorithm to compute the context similarity.

Now we turn to the training phase. $\forall i, (C_i, R_i)$, our goal is to maximize the generative probability of R_i by selecting a prototype $(C'_i, R'_i) \in \mathcal{D}$. As we already know the ground-truth response R_i , we first retrieve thirty prototypes $\{(C'_{i,j}, R'_{i,j})\}_{j=1}^{30}$ based on the response similarity instead of context similarity, and then reserve prototypes whose Jaccard similarity to R_i are in the range of $[0.3, 0.7]$. Here, we use Lucene to index all responses, and retrieve the top 30 similar responses along with their corresponding contexts for R_i . The Jaccard similarity measures text similarity from a bag-of-words view, that is formulated as

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}, \quad (1)$$

where A and B are two bags of words and $|\cdot|$ denotes the number of elements in a collection. Each context-response pair is processed with the above procedure, so we obtain enormous quadruples $\{(C_i, R_i, C'_{i,j}, R'_{i,j})_{j=0}^{M_i}\}_{i=1}^N$ after this step. The motivation behind filtering out instances with Jaccard similarity < 0.3 is that a neural editor model performs well only if a prototype is lexically similar (Gua et al., 2017) to its ground-truth. Besides, we hope the editor does not copy the prototype so we discard instances where the prototype

and groundtruth are nearly identical (i.e. Jaccard similarity > 0.7). We do not use context similarity to construct parallel data for training, because similar contexts may correspond to totally different responses, so-called one-to-many phenomenon in dialogue generation, that impedes editor training due to the large lexicon gap. According to our preliminary experiments, the editor always generates non-sense responses if training data is constructed by context similarity.

4.3 Context-Aware Neural Editor

A context-aware neural editor aims to revise a prototype to adapt current context. Formally, given a quadruple (C, R, C', R') (we omit subscripts for simplification), a context-aware neural editor first forms an edit vector z using C and C' , and then updates parameters of the generative model by maximizing the probability of $p(R|z, R')$. For testing, we directly generate a response after getting the editor vector. In the following, we will introduce how to obtain the edit vector and learn the generative model in details.

4.3.1 Edit Vector Generation

For an unconditional sentence editing setting (Gua et al., 2017), an edit vector is randomly sampled from a distribution because how to edit the sentence is not constrained. In contrast, we should take both of C and C' into consideration when we revise a prototype response R' . Formally, R' is firstly transformed to hidden vectors $\{h_k | h_k = \vec{h}_k \oplus \overleftarrow{h}_k\}_{k=1}^{n_j}$ through a biGRU parameterized as Equation (2).

$$\vec{h}_j = f_{\text{GRU}}(\vec{h}_{j-1}, r'_j); \quad \overleftarrow{h}_j = f_{\text{GRU}}(\overleftarrow{h}_{j+1}, r'_j) \quad (2)$$

²<https://lucenenet.apache.org/>

where r'_j is the j -th word of R' .

Then we compute a context diff-vector $diff_c$ by an attention mechanism defined as follows

$$diff_c = \sum_{w \in I} \beta_w \Psi(w) \oplus \sum_{w' \in D} \gamma_{w'} \Psi(w'), \quad (3)$$

where \oplus is a concatenation operation, $I = \{w | w \in C \wedge w \notin C'\}$ is a insertion word set, and $D = \{w' | w' \in C' \wedge w' \notin C\}$ is a deletion word set. $diff_c$ explicitly encodes insertion words and deletion words from C to C' . β_w is the weight of a insertion word w , that is computed by

$$\beta_w = \frac{\exp(e_w)}{\sum_{w \in I} \exp(e_w)}, \quad (4)$$

$$e_w = \mathbf{v}_\beta^\top \tanh(\mathbf{W}_\beta [\Psi(w) \oplus h_l]), \quad (5)$$

where \mathbf{v}_β and \mathbf{W}_β are parameters, and h_l is the last hidden state of the encoder. $\gamma_{w'}$ is obtained with a similar process:

$$\gamma_{w'} = \frac{\exp(e_{w'})}{\sum_{w' \in D} \exp(e_{w'})}, \quad (6)$$

$$e_{w'} = \mathbf{v}_\gamma^\top \tanh(\mathbf{W}_\gamma [\Psi(w') \oplus h_l]), \quad (7)$$

We assume that different words influence the editing process unequally, so we weighted average insertion words and deletion words to form an edit in Equation 3. Table 1 explains our motivation as well, that is “desserts” is much more important than “the” in the editing process. Then we compute the edit vector z by following transformation

$$z = \tanh(\mathbf{W} \cdot diff_c + \mathbf{b}), \quad (8)$$

where \mathbf{W} and \mathbf{b} are two parameters. Equation 8 can be regarded as a mapping from context differences to response differences.

It should be noted that there are several alternative approaches to compute $diff_c$ and z for this task, such as applying memory networks, latent variables, and other complex network architectures. Here, we just use a simple method, but it yields interesting results on this task. We will further illustrate our experiment findings in the next section.

4.3.2 Prototype Editing

We build our prototype editing model upon a Seq2Seq with an attention mechanism model, which integrates the edit vector into the decoder.

The decoder takes $\{h_k\}_{k=1}^{n_j}$ as an input and generates a response by a GRU language model with

attention. The hidden state of the decoder is acquired by

$$h'_j = f_{\text{GRU}}(h'_{j-1}, r_{j-1} \oplus z_i), \quad (9)$$

where the input of j -th time step is the last step hidden state and the concatenation of the $(j-1)$ -th word embedding and the edit vector obtained in Equation 8. Then we compute a context vector c_i , which is a linear combination of $\{h_1, \dots, h_t\}$:

$$c_i = \sum_{j=1}^t \alpha_{i,j} h_j, \quad (10)$$

where $\alpha_{i,j}$ is given by

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^t \exp(e_{i,k})}, \quad (11)$$

$$e_{i,j} = \mathbf{v}^\top \tanh(\mathbf{W}_\alpha [h_j \oplus h'_i]), \quad (12)$$

where \mathbf{v} and \mathbf{W}_α are parameters. The generative probability distribution is given by

$$s(r_i) = \text{softmax}(\mathbf{W}_p [r_{i-1} \oplus h'_i \oplus c_i] + \mathbf{b}_p), \quad (13)$$

where \mathbf{W}_p and \mathbf{b}_p are two parameters. Equation 11 and 13 are the attention mechanism (Bahdanau et al., 2015), that mitigates the long-term dependency issue of the original Seq2Seq model. We append the edit vector to every input embedding of the decoder in Equation 9, so the edit information can be utilized in the entire generation process.

Since our model is differentiated, we learn our response generation model by minimizing the negative log likelihood of \mathcal{D}

$$\mathcal{L} = - \sum_{i=1}^N \sum_{j=1}^l \log p(r_{i,j} | z_i, R'_i, r_{i,k < j}) \quad (14)$$

We implement our model by PyTorch³. We employ the Adam algorithm (Kingma and Ba, 2015) to optimize the objective function with a batch size of 128. We set the initial learning rate as 0.001 and reduce it by half if perplexity on validation begins to increase. We will stop training if the perplexity on validation keeps increasing in two successive epochs. The word embedding size and editor vector size are 512, and both of the encoder and decoder are a 1-layer GRU whose hidden vector size is 1024. Message and response vocabulary size are 30000, and words not covered by the vocabulary are represented by a placeholder \$UNK\$.

³<https://pytorch.org/>

5 Experiment

5.1 Experiment setting

Our editing model requires a large corpus to find lexically similar prototypes. We crawl over 20 million human-human context-response pairs (context only contains 1 turn) from Douban Group⁴ which is a popular forum in China. After removing duplicated pairs and utterance longer than 30 words, we split 19,623,374 pairs for training, 10,000 pairs for validation and 10,000 pairs for testing. The average length of contexts and responses are 11.64 and 12.33 respectively. We index pairs in the training dataset for prototype retrieval, and our retrieval strategy is described in Section 4.2. We finally obtain 42,690,275 (C_i, R_i, C'_i, R'_i) quadruples. For a fair comparison, we randomly sample 19,623,374 quadruples for training.

5.2 Baselines

S2SA: We adopt Seq2Seq with attention (Bahdanau et al., 2015) as a baseline model. We use a Pytorch implementation, OpenNMT (Klein et al., 2017) in the experiment.

S2SA-MMI: We employed the bidirectional-MMI decoder as in (Li et al., 2016a). The hyperparameter λ is set as 0.5 according to the paper’s suggestion. 200 candidates are sampled from beam search for reranking.

CVAE: The conditional variational auto-encoder is a popular method of increasing the diversity of response generation (Zhao et al., 2017). We use the published code at <https://github.com/snakeztc/NeuralDialog-CVAE>, and conduct small adaptations for our single turn scenario.

Retrieval: We compare our model with retrieval-based methods to show the effect of editing. Here, we use two retrieval strategies. The first one, denoted as **Retrieval-default**, is that we directly use the top-1 result given by Lucene, which ranks candidates with the context similarity. The second one is that we first retrieve 20 responses, and then employ a dual-LSTM model (Lowe et al., 2015) to compute matching degree between current context and response candidates. The matching model is implemented with the same setting in (Lowe et al., 2015), and is trained on the training data set where negative instances are randomly

sampled with a ratio of $pos : neg = 1 : 9$. We call this baseline as **Retrieval-Rerank**.

Correspondingly, we design three variants of our model, **Edit-default**, **Edit-1-Rerank** and **Edit-N-Rerank**. Edit-default and Edit-1-Rerank, edit top-1 response yielded by Retrieval-default and Retrieval-Rerank respectively. Edit-N-Rerank edits all 20 responses returned by Lucene and then reranks the revised results with the dual-LSTM model.

Word embedding size, hidden vector size and attention vector size of baselines and our models are the same. All generative models use beam search to yield responses, where beam size is chosen as 20 except S2SA-MMI. For all models, we remove \$UNK\$ from the target vocabulary, because it always causes fluency issue in evaluation.

5.3 Evaluation Metrics

How to evaluate response generation systematically is still an open problem. We evaluate our model on four criteria, including: fluency, relevance, diversity and originality. We employ Embedding Average (Average), Embedding Extrema (Extrema), and Embedding Greedy (Greedy) (Liu et al., 2016) to evaluate response relevance, which are better correlated with human judgment than BLEU. Following (Li et al., 2016a), we evaluate the response diversity based on the ratios of distinct unigrams and bigrams in generated responses, denoted as Distinct-1 and Distinct-2. In this paper, we define a new metric, originality, that is defined as the ratio of generated responses that do not appear in the training set. Here, “appear” means we can find exactly the same response in our training data set. We randomly select 1,000 contexts from the test set, and ask three native speakers to annotate response fluency. We conduct 3-scale rating: +2, +1 and 0. +2: The response is fluent and grammatically correct. +1: There are a few grammatical errors in the response but readers could understand it. 0: The response is totally grammatically broken, making it difficult to understand.

5.4 Evaluation Results

Table 2 shows the evaluation results on the Chinese dataset. Our methods are better than retrieval-based methods on embedding based metrics, that means revised responses are more relevant to ground-truth in the semantic space. Our model just slightly revises prototype response, so

⁴<https://www.douban.com/group>

Table 2: Automatic evaluation results. Numbers in bold mean that improvement from the model on that metric is statistically significant over the baseline methods (t-test, p-value < 0.01).

	Relevance			Diversity		Originality	Fluency
	Average	Extrema	Greedy	Distinct-1	Distinct-2	Not appear	Avg. Score
S2SA	0.346	0.180	0.350	0.032	0.087	0.208	1.90
S2SA-MMI	0.379	0.189	0.385	0.039	0.127	0.297	1.86
CVAE	0.360	0.183	0.363	0.062	0.178	0.745	1.71
Retrieval-default	0.288	0.130	0.309	0.098	0.549	0.000	1.95
Edit-default	0.297	0.150	0.327	0.071	0.300	0.796	1.78
Retrieval-Rerank	0.380	0.191	0.381	0.067	0.460	0.000	1.96
Edit-1-Rerank	0.367	0.185	0.371	0.077	0.296	0.794	1.79
Edit-N-Rerank	0.386	0.203	0.389	0.068	0.280	0.860	1.78

improvements on automatic metrics are not that large but significant on statistical tests (t-test, p-value < 0.01). Two factors are known to cause Edit-1-Rerank worse than Retrieval-Rerank. 1) Rerank algorithm is biased to long responses, that poses a challenge for the editing model. 2) Despite of better prototype responses, a context of top-1 response is always greatly different from current context, leading to a large insertion word set and a large deletion set, that also obstructs the revision process. In terms of diversity, our methods drop on distinct-1 and distinct-2 in a comparison with retrieval-based methods, because the editing model often deletes special words pursuing for better relevance. Retrieval-Rerank is better than retrieval-default, indicating that it is necessary to rerank responses by measuring context-response similarity with a matching model.

Our methods significantly outperform generative baselines in terms of diversity since response prototypes are diverse and informative. It demonstrates that the prototype-then-editing paradigm is essentially capable of addressing the safe response problem. Edit-Rerank is better than generative baselines on relevance but Edit-default is not, indicating a good prototype selector is quite important to our editing model. In terms of originality, about 86% revised response do not appear in the training set, that surpasses S2SA, S2SA-MMI and CVAE. This is mainly because baseline methods are more likely to generate safe responses that are frequently appeared in the training data, while our model tends to modify an existing response that avoids duplication issue. In terms of fluency, retrieval based approaches achieve the best results, and S2SA comes to the second place. Safe response enjoys high score on fluency, that is why

S2SA and S2SA-MMI perform well on this metric. The edit model obtains an average score of 1.79. That is an acceptable fluency score for a dialogue engine and most of generated responses are grammatically correct.

5.5 Discussions

5.5.1 Editing Type Analysis

It is interesting to explore the semantic gap between prototype and revised response. We ask annotators to conduct 4-scale rating on 500 randomly sampled prototype-response pairs given by Edit-default and Edit-N-Rerank respectively. The 4-scale is defined as: identical, paraphrase, on the same topic and unrelated.

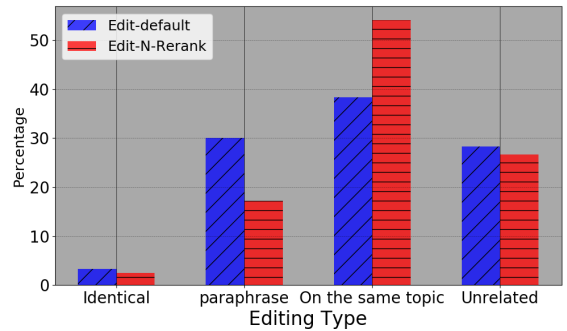


Figure 2: Editing Ratio. The different between paraphrase and on the same topic is that a response in “on the same topic” category introduces or deletes contents to its prototype response.

Figure 2 provides the ratio of four editing types defined above. For both of two methods, Only 2% of edits are exactly the same with the prototype, that means our model does not downgrade to a copy model. Surprisingly, there are 30% revised responses are unrelated to prototypes. The key factor for this phenomenon is that the neural

editor will rewrite the prototype when it is hard to insert insertion words to the prototype. The ratio of “on the same topic” response given by Edit-N-rerank is larger than Edit-default, revealing that “on the same topic” responses might be more relevant from the view of a LSTM based reranker.

5.5.2 Case Study

We give three examples to show how our model works in Table 3. The first case illustrates the effect of word insertion. Our editing model enriches a short response by inserting words from context, that makes the conversation informative and coherent. The second case gives an example of word deletion, where a phrase “braised pork rice” is removed as it does not fit current context. Phrase “braised pork rice” only appears in the prototype context but not in current context, so it is in the deletion word set D , that makes the decoder not generate it. The third one is that our model forms a relevant query by deleting some words in the prototype while inserting other words to it. Current context is talking about “clean tattoo”, but the prototype discusses “clean hair”, leading to an irrelevant response. After the word substitution, the revised response becomes appropriated for current context.

According to our observation, word deletion and replacement operations have a positive effect on relevance. Specifically, word deletion operation is able to remove words that do not fit to current context, which makes a response coherent to current context. Similarly, word replacement operation adds context related words to prototype while remove unappropriated ones.

6 Conclusion and Future Work

We present a new paradigm, prototype-then-edit, for open domain response generation, that enables a generation-based chatbot to leverage retrieved results. We propose a simple but effective model to edit context-aware responses by taking context differences into consideration. Experiment results on a large-scale dataset show that our model improves the prototype relevance, and outperforms other generative baselines in terms of both of relevance and diversity. In the future, we will extend our model to multi-turn response generation, that considers complex relationship among context utterances. We will also investigate how to learn prototype selector and neural editor in an end-to-end manner.

Table 3: Case Study. We show examples yielded by Edit-default and Edit-Rerank. Chinese utterances are translated to English here.

Insertion case	
Context	身 在 国 外 寂 寞 无 聊 就 化 妆 When I feel bored and lonely abroad, I like makeup.
Prototype context	无 聊 就 玩 If I feel bored, I will play something.
Prototype response	嗯 OK.
Revised response	嗯 我 也 喜 欢 化 妆 OK. I love makeup too.
Deletion case	
Context	我 比 较 常 吃 辛 拉 面 I often eat spice noodles.
Prototype context	我 在 台 湾 时 候 常 吃 辛 拉 面 和 卤 肉 饭。 When I lived at Taiwan, I often eat spicy Noodles and braised pork rice.
Prototype response	我 也 喜 欢 卤 肉 饭。 I love braised pork rice as well.
Revised response	我 也 喜 欢。 I love it as well. (In Chinese, model just deletes the phrase “braised pork rice” without adding any word.)
Replacement case	
Context	纹 身 有 没 有 办 法 全 部 弄 干 净 Is there any way to get all tattoos clean
Prototype context	把 药 抹 头 发 上 能 把 头 发 弄 干 净 么 Is it possible to clean your hair by wiping the medicine on your hair?
Prototype response	抹 完 真 的 头 发 干 净 很 多 After wiping it, hair gets clean much.
Revised response	抹 完 纹 身 就 会 掉 很 多 After wiping it, most of tattoos will be cleaned.

References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR*.
- SL Ziqiang Cao, Wenjie Li, Furu Wei, and Sujian Li. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *ACL*, volume 2, page 3.
- Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

- Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2017. [Generating sentences by editing prototypes](#). *CoRR* abs/1709.08878. <http://arxiv.org/abs/1709.08878>.
- Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050.
- Zongcheng Ji, Zhengdong Lu, and Hang Li. 2014. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*.
- Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR*.
- Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. [Opennmt: Open-source toolkit for neural machine translation](#). In *Proc. ACL*. <https://doi.org/10.18653/v1/P17-4012>.
- Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 110–119.
- Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016b. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1192–1202.
- Jiwei Li, Will Monroe, Tianlin Shi, Alan Ritter, and Dan Jurafsky. 2017. Adversarial learning for neural dialogue generation. *EMNLP*.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. *NAACL*.
- Yi Liao, Lidong Bing, Piji Li, Shuming Shi, Wai Lam, and Tong Zhang. 2018. Incorporating pseudo-parallel data for quantifiable sequence editing. *arXiv preprint arXiv:1804.07007*.
- Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *EMNLP*.
- Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. 2017. [Towards an automatic turing test: Learning to evaluate dialogue responses](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1116–1126. <https://doi.org/10.18653/v1/P17-1103>.
- 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. *SIGDIAL*.
- Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2016. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. *arXiv preprint arXiv:1607.00970*.
- Alan Ritter, Colin Cherry, and William B Dolan. 2011. Data-driven response generation in social media. In *EMNLP*, Association for Computational Linguistics, pages 583–593.
- Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. End-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, 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 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1577–1586.
- Yiping Song, Rui Yan, Xiang Li, Dongyan Zhao, and Ming Zhang. 2016. Two are better than one: An ensemble of retrieval-and generation-based dialog systems. *arXiv preprint arXiv:1610.07149*.
- Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 196–205.
- Jinyue Su, Jiacheng Xu, Xipeng Qiu, and Xuanjing Huang. 2018. Incorporating discriminator in sentence generation: a gibbs sampling method. *arXiv preprint arXiv:1802.08970*.
- Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

- Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. 2018. RUBER: an unsupervised method for automatic evaluation of open-domain dialog systems. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.
- Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869*.
- Di Wang, Nebojsa Jojic, Chris Brockett, and Eric Nyberg. 2017. Steering output style and topic in neural response generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. pages 2140–2150.
- Joseph Weizenbaum. 1966. Eliza: a computer program for the study of natural language communication between man and machine. *Communications of the ACM* 9(1):36–45.
- Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. [Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots](#). In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. pages 496–505. <https://doi.org/10.18653/v1/P17-1046>.
- Yu Wu, Wei Wu, Dejian Yang, Can Xu, and Zhoujun Li. 2018. Neural response generation with dynamic vocabularies. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.
- Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*. pages 1782–1792.
- Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *AAAI 2017*. 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. *AAAI-18*.
- Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie Sun, Xiaolong Wang, Zhuoran Wang, and Chao Qi. 2017. Neural response generation via GAN with an approximate embedding layer. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. pages 617–626.
- Rui Yan, Yiping Song, and Hua Wu. 2016. Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. pages 55–64.
- Lili Yao, Yaoyuan Zhang, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017. Towards implicit content-introducing for generative short-text conversation systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. pages 2190–2199.
- Tiancheng Zhao, Ran Zhao, and Maxine Eskénazi. 2017. [Learning discourse-level diversity for neural dialog models using conditional variational autoencoders](#). In *ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. pages 654–664. <https://doi.org/10.18653/v1/P17-1061>.
- Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2017. Emotional chatting machine: Emotional conversation generation with internal and external memory. *arXiv preprint arXiv:1704.01074*.
- Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. pages 372–381.