Simultaneously Improving Utility and User Experience in Task-oriented Dialogue Systems

Phillip Lippe
University of Amsterdam
p.lippe@uva.nl

Pengjie Ren
Shandong University
renpengjie@sdu.edu.cn

Hinda Haned
University of Amsterdam
h.haned@uva.nl

Bart Voorn
STRM Privacy
bartvoorn@gmail.com

Maarten de Rijke
University of Amsterdam
m.derijke@uva.nl

ABSTRACT
Task-oriented dialogue systems (TDSs) help users achieve a specific task through conversations, e.g., in grocery shopping or at help desks. Dialogue response generation (DRG) is a core TDS component that translates system actions into natural language responses. Methods for DRG in TDSs tend to be template-based or corpus-based. The former fill slots in templates with system actions to produce responses at run-time. The latter generate responses token by token by taking system actions into account. In an e-commerce setting, both approaches have strengths and weaknesses: (i) template-based DRG provides high precision and highly predictable responses but may fail to generate diverse and natural responses, thus hurting the user experience; and (ii) corpus-based DRG is able to generate natural responses but its precision or predictability cannot be guaranteed, thus hurting the utility.

To improve the user experience of conversational interactions without hurting utility we introduce P2-Net, a prototype-based, paraphrasing neural network. P2-Net enhances the precision and diversity of responses. Instead of generating a response from scratch, P2-Net generates system responses by paraphrasing template-based responses. To guarantee precision, P2-Net learns to separate a response into its semantics, context influence, and paraphrasing noise, and to keep the semantics unchanged during paraphrasing. To boost diversity, P2-Net samples previous conversational utterances as prototypes, from which it can then extract speaking style information.

We conduct experiments on the MultiWOZ dataset with automatic and human evaluations. P2-Net achieves a significant improvement in diversity while preserving the semantics of responses.

ACM Reference Format:

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGIR eCom’22, July 15, 2022, Madrid, Spain
© 2022 Copyright held by the owner/author(s).

1 INTRODUCTION
Task-oriented dialogue systems (TDSs) have become widespread in e-commerce, e.g., with uses as shopping assistant, at help desks, and in customer service [9, 23, 24, 38, 53]. Two key factors contribute to overall user satisfaction with TDSs, utility and user experience [39, 42]. In current approaches to dialogue response generation (DRG), a core TDS component, these two factors are often addressed in one of two ways [5]. Template-based approaches to DRG use manually created response templates, which are instantiated with slot values at run-time. They tend to produce high-precision results with a high degree of predictability but have a low degree of diversity, which may result in unnatural conversations, thus hurting the user experience. In contrast, corpus-based approaches to DRG directly generate responses token by token at run-time and thereby generate responses that tend to be diverse and fluent, but they may generate unexpected responses.

Figure 1: Overview of a combined template-based and corpus-based approach to response generation. First, a template-based dialogue system generates a template response based on the user’s question. Second, the response is refined by a paraphrasing model that takes the conversational context into account.

How can we generate responses that are both more useful and more engaging? We propose to refine responses produced by a template-based system with a corpus-based model based on a combination of neural prototype editing [13] and paraphrasing techniques [18]. We assume that a response is generated subject to three high-level constraints: the semantics (i.e., what to say), the context style (i.e., the user’s question and previous dialogue turns), and paraphrasing noise (i.e., unnecessary words, rephrasing). The semantics can
be determined best by a template-based approach. The context style and paraphrasing noise must be flexible as there are many ways of expressing the same meaning, e.g., using different sentence functions; hence, a corpus-based approach is best suited for these components. By rephrasing the template-based response with a corpus-based model, we want to keep the high controllability and precision of a template-based approaches (thus ensuring utility), while generating more diverse responses and natural conversations as in corpus-based approaches to response generation (thus improving the user experience). In this manner we seek to satisfy the two key constraints TDSs need to meet in an e-commerce context.

Fig. 1 illustrates the combined strategy that we propose. The combined strategy is significantly simpler than generating a response from scratch, thereby allowing us to focus on style details.

This task differs from previous work on diversifying text generation through style transfer [7, 37], which aims to rewrite a sentence with a target style, while keeping the semantics mostly unchanged. In our task, it is not sufficient to simply adjust to the style of the user because we need to establish a natural conversation with filling words like ‘sure’ or ‘of course’. It also differs from traditional paraphrasing [27, 50] as we should not just diversify the templates, but also incorporate the conversational context.

To operationalize the process in Fig. 1, we propose a prototype-based, paraphrasing neural network, called P2-Net. P2-Net learns to encode the three response components independently, i.e. semantics, context style, and paraphrasing noise. We strongly limit the information flow from the ground truth response, ensuring that the ground truth response can only help to extract latent style information (i.e., the paraphrasing noise) from response style prototypes that it cannot retrieve from the other sources.

P2-Net is trained on the task of generating the ground truth responses. As no sufficiently large dataset of aligned template-based and corpus-based responses exists, we propose a weakly-supervised learning mechanism to train P2-Net, where we assume system responses with the same system actions are paraphrases with the same semantics but different styles. We compare P2-Net to stochastic beam search (an effective method to promote diverse responses w.r.t. different prototype sentences. The idea of paraphrasing is to rephrase a sentence in different styles without changing its semantics [30, 50]. We use paraphrasing to make sure that the semantics of the rephrased is kept unchanged. To the best of our knowledge, no prior work has proposed to boost the user experience while maintaining utility in TDSs.

2 RELATED WORK

User satisfaction with TDSs. While TDSs are typically optimized for utility, as determined, e.g., in terms of task completion or conversion, there is growing awareness that two key dimensions determine overall user satisfaction with TDSs: utility and user experience [39].

Dialogue response diversity. Diversifying the responses produced by conversational agents is a topic of growing interest [15, 19, 36, 49, 52]. There have been many approaches to reach this goal. One is to adjust the loss function or learning mechanism to encourage diversity [15, 19]. While these methods increase token diversity, they might promote other diversity aspects like sentence structure or phrasal paraphrasing. Some studies adopt generative adversarial networks [11], where the discriminator is used to distinguish between real and fake samples [20, 49]. While this approach has been shown to generate more human-like responses, training can be very unstable and may not boost the results as much as expected [20]. Besides, the methods listed above have all (initially) been proposed for chitchat and cannot be applied to TDS directly, as they cannot guarantee to preserve the semantics of the responses [33].

The diversity of response generation has been widely studied in open-domain dialogue systems, where a commonly used approach is beam search [43], which diversifies responses by changing the way one samples each token from each decoding step [36, 45]. These methods can be applied to any already trained sequence-to-sequence generation models. However, the diversity of response generation has not been investigated in TDSs yet, including beam search based methods. In this work, we compare our proposed new workflow to beam search based methods when applied to TDSs.

Paraphrasing. Paraphrasing refers to the task of detecting and generating paraphrases. Conventional approaches model paraphrase generation as a supervised encoding-decoding process [12, 29]. Some work uses deep reinforcement learning approaches to paraphrase generation [21, 30]. Other studies investigate weakly-supervised paraphrasing by synthesizing pseudo-paraphrase pairs [17, 48]. There are also unsupervised paraphrasing studies [2]. E.g., Liu et al. [22] model paraphrase generation as an optimization problem and consider semantic similarity, expression diversity, and language fluency to define the learning objective. Paraphrasing has also been applied to boost the performance of tasks such as machine translation [1], information retrieval [54], and dialogue systems [10, 35].

What we add on top of the work discussed above is a new schema to diversify DRG in TDSs based on prototype editing and paraphrasing. The idea of prototype editing is to first sample a prototype sentence from the training corpus and then edit it into a new sentence, instead of generating a sentence from scratch [13]. The prototype sentences have different styles so that we expect to get diverse responses w.r.t. different prototype sentences. The idea of paraphrasing is to rephrase a sentence in different styles without changing its semantics [30, 50]. We use paraphrasing to make sure that the semantics of the rephrased is kept unchanged. To the best of our knowledge, no prior work has proposed to boost the user experience while maintaining utility in TDSs.

3 METHOD

Given a template response (from a template-based TDS system) and a dialogue context (from previous turns), the task is to paraphrase
the template response to (i) keep its semantics unchanged, and (ii) increase its diversity. For (i), we need to ensure all slots of the template response are covered and placed in the right position of the response. For (ii), we need to make the response aware of context and incorporate random noise that can only influence the non-essential content of the response.

3.1 Overview of P2-Net

We assume that three main factors contribute to a response of a TDS being human-like: (i) the semantics, (ii) context style, and (iii) paraphrasing noise. The semantics of a response determines the message to communicate to the user, and template-based TDSs perform especially well on it. There are various ways to express the same semantics. It is influenced by the context style, i.e., the preceding conversation and the question of the user. Depending on the specific way the user is asking their question, we can respond more naturally. E.g., if the question is ‘Can you tell me the name of the hotel?’, the TDS could respond with ‘I absolutely can, the name is . . . ’ while this starting phrase is not suitable for all questions. Even if the context turns some of the paraphrases inappropriate, there may be sentence variations, which we summarize as paraphrasing noise, i.e., redundant words like ‘sure’ and ‘of course.’

We propose our context-aware paraphrasing model, P2-Net; see Fig. 2. The input template response is encoded by a Bi-LSTM into a response semantic vector (1) in Fig. 2 constituting a feature vector. The context style vector (2) and paraphrasing noise vector (3) are represented by modeling context prototypes and response prototypes from which the model can select a weighted sum. All three vectors are input to the decoder (4). The goal is to generate diverse responses while being able to alternate the style without changing the semantics. To learn the split between semantics, context style, and paraphrasing noise, the model is trained to predict the next response in a conversation given different inputs for each of the components. The semantics of the response is modeled by encoding the output of a template-based TDS for the corresponding conversation (1). The context component is extracted from previous conversation turns by the user and the TDS (2). Paraphrasing noise cannot easily be predicted on external inputs as it is based on random choice. We therefore propose to model it from the ground truth directly while limiting the information flow to prevent P2-Net from simply copying the response (3).

During training, P2-Net learns to generate responses based on a template, a dialogue context, and the ground truth. After training, we replace the ground truth with a sampling mechanism to obtain paraphrasing noise inputs, where we sample the attention distribution, which is used for creating the weighted sum over prototypes, by a Dirichlet prior. This setup is expected to generate more diverse outputs than post-processing methods such as beam search because the model explicitly learns different styles of paraphrasing.

3.2 Embeddings

We use two types of embedding: word embeddings and slot embeddings. For the word embeddings, we use GloVe [28] as initialization and fine-tune them during training.

A template from the template-based TDS provides slots in which specific information such as restaurant names or phone numbers are stored. Paraphrasing a template requires an understanding of these slots, and hence they should be taken differently as word embeddings and need to be properly embedded in the neural model. To represent the given slots in a template, three components are necessary. The approach is visualized in Fig. 3.
First, to recognize the general semantics of a slot, we learn an embedding for each type (e.g., area, name, etc.). Second, we distinguish between slots with the same type in case we have a template with, for example, multiple restaurant names. The order of slots can be important as well: if we have two names and two addresses, the network needs to reason about which name belongs to which address. To implement this ordering, we use a sinusoidal position embedding [44]. Third, the actual value of the slot is relevant to form a natural sentence. We choose a simple approach to embed the values, namely a single-layer Continuous Bag-of-Words (CBOW) with a gate modeled by the slot type embedding. The CBOW prevents strong overfitting on the slot values, and the gate controls how much information is necessary to improve the slot representation. All three components combined result in the final representation that is used in the encoder and decoder.

3.3 Encoder
The encoder generates semantic and style representations; each has a specific architecture.

Semantic encoding. The template response (\(1\)) is used to encode the semantics by using a one-layer Bi-LSTM [14] network with global attention, using the last hidden state \(h_{\text{end}}\). The attention can be specified as follows:

\[
\hat{p}_{\text{semantics}} = \frac{\sum_{t=1}^{T} h_t \cdot \exp(\text{attn}(h_t; h_{\text{end}}))}{\sum_{t=1}^{T} \exp(\text{attn}(h_t; h_{\text{end}}))}
\]

\[
\text{attn}(h^{(i)}; h^{(T)}) = \text{tanh}(W_k h^{(i)} + W_c h^{(T)}) + b_{\text{attn}},
\]

where \(h_t\) is the hidden state of the Bi-LSTM at timestep \(t\). We refer to the output feature vector, \(\hat{p}_{\text{semantics}}\), as response semantic vector, and the attention distribution as semantic attention of a response.

Context style encoding. The context style is encoded with a hierarchical RNN on a limited number of previous conversation turns (\(2\)). We use the same one-layer Bi-LSTM network as for semantic encoding, but with an attention module with separate weights, which we refer to as context style attention. We devise a prototype layer by introducing a fixed set of learnable embeddings, which we call context prototypes \(p^c_1,\ldots,p^c_K\) (\(2\)). The context style vector is a weighted sum of these prototypes. The weights are determined by the context using an attention module:

\[
\hat{p}_{\text{style}}^c = \frac{\sum_k p^c_k \cdot \exp(\text{attn}(p^c_k; \hat{p}_{\text{context}}))}{\sum_k \exp(\text{attn}(p^c_k; \hat{p}_{\text{context}}))},
\]

where \(p^c_k\) are the context prototype vectors and \(\hat{p}_{\text{context}}\) is the encoded feature vector of the context (the last hidden state of LSTM). The prototype layer prevents the model from encoding a significant amount of unnecessary information into the feature vector, and thus might help to generalize better.

3.3.1 Paraphrasing noise encoding. As explained previously, paraphrasing noise can only be determined by the ground truth response. During training, we encode the ground truth into a feature vector representing the response paraphrasing noise (\(3\)). We use the same one-layer Bi-LSTM as for template encoding, but with an attention module with separate weights, which we refer to as paraphrasing noise attention. We create a bottleneck to limit the information flow from the ground truth. Again, we do this by introducing a fixed set of response prototypes \(p^r_1,\ldots,p^r_K\) (\(3\)). The response paraphrasing noise vector is a weighted sum of these prototypes. The weights are determined by another attention module:

\[
\hat{p}_{\text{noise}} = \frac{\sum_k p^r_k \cdot \exp(\text{attn}(p^r_k; \hat{p}_{\text{response}}))}{\sum_k \exp(\text{attn}(p^r_k; \hat{p}_{\text{response}}))},
\]

where \(p^r_k\) are the prototype vectors and \(\hat{p}_{\text{response}}\) is the encoded feature vector of the ground truth (the last hidden state of the LSTM). The ground truth can guide the generation process by providing information that cannot be extracted from the template and context, but not enough for the generation process to fully reconstruct the response solely from this representation.

During evaluation, the ground truth response is not available. To obtain diverse responses, we sample from the paraphrasing noise attention, e.g., with a Dirichlet distribution. Different sampling results (combinations of response prototypes) should lead to different responses with the same semantics, but expressed differently.

3.4 Decoder
Based on the semantics \(\hat{p}_{\text{semantics}}\), the context style vector \(\hat{p}_{\text{style}}^c\), and the response paraphrasing noise vector \(\hat{p}_{\text{noise}}\), the decoder generates a new response specific to the inputs. The module is inspired by the pointer network architecture [34, 46], and consists of a one-layer unidirectional LSTM as base network. See Fig. 4.

The initial state is generated based on the encoded context style and semantics. We use the current state \(h_t\) as context vector to determine an attention distribution \(p_{\text{slot}}\) over the slots that should be included in the output response. The weighted sum of the slot embeddings (see §3.2) is used as an additional input for determining the output distribution \(p_{\text{word}}\) over words. Furthermore, a binary classifier is applied to determine whether the next word should be generated from the vocabulary (\(p_{\text{gen}} = 1\)), or a slot should be used instead (\(p_{\text{gen}} = 0\)). The probability is calculated as follows:

\[
p_{\text{gen}} = \sigma\left(w_h h^D + w_c \hat{p}_{\text{semantics}} + w_s \hat{p}_{\text{style}} + w_n \hat{p}_{\text{noise}} + b_{\text{gen}}\right),
\]

where \(h^D\) is the hidden state of the decoder at timestep \(t\); and \(\hat{p}_{\text{semantics}}, \hat{p}_{\text{style}}\) and \(\hat{p}_{\text{noise}}\) are the encoded semantics, context style vector and the paraphrasing noise vector, respectively.

During inference and sampling, we experienced that obtaining a probability distribution over all tokens, i.e., multiplying \(p_{\text{gen}}\) with
the probabilities over the vocabulary and $1 - p_{\text{gen}}$ with the attention distribution over the slots, strongly favors the slots. To counteract this behavior, we generate a new word if $p_{\text{gen}} > \delta$, and otherwise select a slot; we set $\delta = 0.5$ for stable and good results.

Another important aspect of the slots is that in most responses, each slot is only used once in a prediction. In our dataset (see §4), we experienced that almost 99% of the answers given by a human contained each slot only once. Therefore, we expect the network to learn using each slot once as well. It might be hard for the decoder to remember whether it has already used a certain slot, which may lead to repetitive outputs. To prevent this, we introduce an inductive bias by masking out slots that have already been used in the output. During training, we mask slots based on the ground truth, while for inference, we do it when the network predicts a slot.

### 3.5 Learning

Given a conversational context, a template, and a ground truth response, we train P2-Net to reconstruct the ground truth response. We consider responses with the same dialogue action and the same slots (types and amount, not actual values) as paraphrases in different contexts. So for a given ground truth response, its paraphrases are considered as templates. Let $y(i)$ denote whether the token at position $i$ of the ground truth response is a slot ($y(i) = 0$) or a word ($y(i) = 1$). Then, the loss for binary classifier $p_{\text{gen}}$ is defined as:

$$L_{\text{gen}} = -\sum_i y(i) \log p_{\text{gen}}^{(i)} + (1 - y(i)) \log (1 - p_{\text{gen}}^{(i)}).$$

(5)

If $y(i) = 0$, i.e., the token is a slot, we add the negative log likelihood of that slot in the decoder’s attention distribution $p_{\text{slot}}$. In case $y(i) = 1$, i.e., the token is a word, we add the negative log likelihood of the word in the decoder’s output distribution $p_{\text{word}}$:

$$L_{\text{word}}^{(i)} = \begin{cases} -\log p_{\text{slot}}^{(i)} & \text{if } y(i) = 0 \\ -\log p_{\text{word}}^{(i)} & \text{if } y(i) = 1. \end{cases}$$

(6)

The final loss is a combination of $L_{\text{gen}}$ and $L_{\text{word}}^{(i)}$:

$$L_{\text{final}} = L_{\text{gen}} + \sum_i L_{\text{word}}^{(i)}.$$  

(7)

### 4 EXPERIMENTAL SETUP

We seek to answer the following research questions: (RQ1) Can P2-Net generate more diverse responses than post-processing methods? And which variant of P2-Net performs best? (RQ2) Is P2-Net able to paraphrase a template without changing its semantics? (RQ3) Can P2-Net learn to attend to tokens w.r.t. semantics with semantic attention and tokens w.r.t. speaking styles with context style attention? (RQ4) How diverse are the responses of P2-Net demonstrated with qualitative analysis? What are typical failures?

**Dataset.** To ensure that we train on human responses that fit the context and have natural conversations, we require dialogues between two humans where one replaces the automated dialogue system. We perform our experiments on the MultiWOZ dataset [4], which contains human-to-human conversations across multiple domains relevant to our e-commerce context. Every response is annotated with a dialogue action and slot entities (e.g., the name of a hotel) used in the sentence. To obtain our templates, we group responses with the same dialogue action and the same slots (types and amount, not actual values) as paraphrases. In this set, we can use any sentence to represent the template for another sentence as they are expected to have the same semantics. If a response has more than one sentence and/or dialogue action, we split it to prevent a mixture of multiple semantics. To counteract overfitting, we only consider response sets with at least four responses. This yields 1,147 sets of different dialogue actions and/or slots, and about 68,000 responses.

Certain sets contain many more responses than others, as, e.g., the dialogue action ‘general request more’ has over 12,000 instances. To prevent the model from focusing only on those responses, we balance the training set by controlling the frequency with which examples from a dialogue action are shown. We take a frequency proportional to the square root of the number of instances for a dialogue action, with an upper limit of 200.

The validation and test datasets are built from 100 response sets for which the network has seen examples (but different contexts and responses), and 100 sets with a new, unseen dialogue action. All sets have 5–7 responses. Hence, we test whether systems can generalize to new contexts and dialogue actions/template semantics.

**Baselines.** Diversity of dialogue response generation has not been investigated in TDSs yet, to the best of our knowledge. Thus, we cannot find prior methods from TDSs for a fair comparison. In open domain dialogue systems, beam search is the commonly used approach to diversifying responses. As a baseline, we perform beam search on the output. Standard beam search gives less diverse results [45], and extensions like stochastic beam search [36] have been proposed instead. For us, the best beam search method was stochastic beam search, possibly due to its sampling behavior, which also introduces diversity by incorporating random noise.

To further set up a baseline, we train P2-Net with two configurations: (i) P2-Net with context and slots as inputs, and (ii) P2-Net with context, slots and template as inputs. For these two configurations, we do not use the context style prototypes and prototype layer is thereby removed and the context style prototypes, which represents standard setups for response generation on the MultiWOZ dataset, except that we provide the slots and/or templates to include in the response instead of a database [4].

**Diversity evaluation.** A commonly used metric for diversity is Distinct-$n$ (or Dist.-$n$ for short), the proportion of unique unigrams/bigrams compared to the overall sentence lengths [19]:

$$\text{Distinct-}n = \frac{\sum_{n=1}^{N} |W_n|}{\sum_{n=1}^{N} |W_n|},$$

where $|W_n|$ denotes the set of uni- or bigrams in the sample $n$, and $|W_n|$ the number of elements in this set. We view each slot as a single token, independent of the size of its content.

**Semantic evaluation.** Besides diversity, it is important to evaluate coherence and textual correctness of generated responses. Diversity can be maximized by learning a uniform distribution over words, but such responses are obviously not useful. We evaluate the semantics of our responses either (i) automated or (ii) through human evaluation.

**Automated evaluation.** For automated evaluation, we use the BLEU metric [26] on the generated responses of the test set. BLEU has been shown to correspond reasonably well with human judgments on this task [8]. We evaluate the BLEU score for both the
responses generated if no ground truth is used as input, i.e., the GT style vector set to zero, and if it is actually used. The second score indicates how much the model relies on the ground truth.

**Human evaluation.** We performed a human evaluation, where a human assessor is presented with a conversation and six generated responses for the last action. The responses had to be evaluated based on four metrics: **Grammaticality**, **Naturalness**, **Context awareness** and **Semantic correctness**. **Grammaticality** judges the English grammar and sentence structure. **Naturalness** measures how ‘human-like’ a response appears to be. **Context awareness** captures whether the generated responses fit into the conversation or not. Lastly, we want to ensure that the semantics of the template response is left unchanged which is judged by the **Semantic correctness**. Ideally, responses of different styles still communicate the same message. For this metric, we also provide the ground truth response from the human agent in the MultiWOZ dataset.

**Implementation details.** We use Adam [16] with a learning rate of 1e-4 and dropout [41] with a rate of 0.2 throughout the network. We start training with a teacher forcing ratio of 0.95, and reduce it exponentially to reach 0.8 after 50k iterations. The hidden size of the LSTMs and the response semantic size is 512. For the context and response, we use four prototypes each and a size of 256 and 64, respectively. We sample $N = 8$ times for every instance in the test dataset by alternating the prototype distribution of P2-Net; we sample the attention distribution by a Dirichlet prior with $\alpha = 0.25$; the template, slots and context are kept fixed for all 8 generated responses. We keep the size of the ground-truth influenced style small so as to bias the network to focus on the context.

We want the ground truth to be considered as ‘extra’ information and not necessary to generate a valid, grammatical response. We use a two-step dropout strategy to augment the response paraphrasing noise vector during training. In 40% of the cases, we set the response paraphrasing noise vector to 0. For the remaining 60%, we sample from a geometric distribution with $p = 0.4$ to determine until which generation time step we set the response paraphrasing noise to 0. Hence, in $p = 40\%$ of the cases we set the paraphrasing noise to 0 for the first 0 steps. Similarly, in $1 - p = 24\%$ of the cases, we set it to zero only for generating the first token, and so on.

## 5 RESULTS AND ANALYSIS

### 5.1 Performance in terms of diversity

To address RQ1, we compare variants of P2-Net with a stochastic beam search. The variants of P2-Net are different combinations of the following inputs: (1) Context: previous dialogue utterances. (2) Slots: slots that should be included in the final response. (3) Template: sampled response template that is used by P2-Net to extract style information. (4) GT: ground truth response, only used during training. (5) Context (proto): context with applied prototype layer.

The evaluation results are listed in Table 1.

First, in terms of diversity, P2-Net outperforms the stochastic beam search baseline by a large margin. Specifically, Dist-2 is improved by around 0.3 while Dist-1 is improved by around 0.15. Stochastic beam search significantly improves beam search [36]; it achieves around 1.5 times more distinct unigrams and up to 3 times more distinct bigrams per sentence compared to standard beam search. This means that stochastic beam search is a strong method in terms of diversifying response generation. P2-Net outperforms stochastic beam search by a large margin, which means that P2-Net generates more diverse responses than post-processing methods like stochastic beam search. A major drawback we experienced with stochastic beam search is that its diversity decreases over training iterations. The longer we train, the lower the diversity of stochastic beam search. In contrast, for P2-Net diversity increases over time.

Second, the variants (4)–(6) achieve comparable performance in terms of diversity. GT + Slots + Template achieves the best performance in terms of both Distinct-1 and Distinct-2. When using the context prototypes as inputs, the diversity performance drops a bit. The model needs to take into account the coherence with context through context prototypes by generating some context-aware words, which will hurt diversity a little bit. E.g., for a context utterance starting with ’Can you . . . ’, it will usually generate responses starting with ’Okay’ or ’Sure’. Interestingly, the performance drops a lot when using the original context instead of context prototypes. When investigating the context style attention distributions on the context utterances, we see that the model focuses on names or specific times. Using context prototypes solved this problem. The training loss is significantly lower than that of the model with prototypes. This indicates that the model overfits on specific contexts, and pays less attention to the ground truth style vector.

### 5.2 Performance in terms of semantics

Turning to RQ2, although P2-Net achieves significant improvements in terms of generating diverse responses, this does not necessarily imply that P2-Net can create a better user experience in practical systems. In an extreme case, we can randomly select/generate responses to get near perfect diversity metrics, but the responses are useless because they lack semantic coherence, which cannot help users to achieve their task goals. To this end, we also conduct experiments to evaluate the semantics of the generated responses.

We report on automatic evaluation using BLEU to check the overlap between generated responses and the demonstrated ground truth responses. See Table 1. P2-Net gets comparable results by using prototypes guided paraphrasing. Specifically, variant (3), Context + Slots + Template, is the baseline here without using prototypes or incorporating paraphrasing noise. By adding the context prototypes, we see that the BLEU score of variant (4) drops only 0.51%, which is acceptable. Also, by further adding the paraphrasing
which means they achieve comparable performance in terms of Context awareness. We believe the reason is that there are a number of cases where the human written responses seem to be more diverse in terms of speaking styles. E.g., P2-Net attends mostly on the slot values, which represent the semantics of the sentences. In the first example, the number of places "wagamama" gets the most attention. Conversely, the other tokens are mostly ignored, so P2-Net indeed tries to extract semantics from template responses.

Second, although both P2-Net and stochastic beam search get satisfactory results, both perform worse than Human Responses. This confirms the reliability and trustworthiness of the human evaluation results in Table 2. For all the experiments in this paper, we assume that the correct system actions (slot and values to be included in the responses) are provided, which makes it easier for the model to generate the responses. In a practical system, this is usually achieved by a natural language understanding module and/or a dialogue policy module. Since we only target the response generation module, we assume slot and values are given beforehand. In practice, even the template-based systems may give improper or incomplete system actions, so we would expect even worse performance than Human Responses in real systems. An exception is that P2-Net gets better performance than Human Responses in terms of Context awareness. We believe the reason is that there are a number of cases where the human written responses seem to be more diverse in terms of speaking styles. E.g., P2-Net.

<table>
<thead>
<tr>
<th>Metric</th>
<th>P2-Net vs SBS</th>
<th>P2-Net vs HR</th>
<th>SBS vs HR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grammaticality</td>
<td>93</td>
<td>100</td>
<td>104</td>
</tr>
<tr>
<td>Naturalness</td>
<td>104</td>
<td>104</td>
<td>104</td>
</tr>
<tr>
<td>Semantic corr.</td>
<td>104</td>
<td>104</td>
<td>104</td>
</tr>
<tr>
<td>Context awar.</td>
<td>105</td>
<td>104</td>
<td>104</td>
</tr>
</tbody>
</table>

Figure 5: Context style attention and semantic attention visualization. Lighter color means higher attention weights.

5.3 Style and semantic attention

To see whether P2-Net can correctly extract style and semantic information from prototypes and template, respectively, we visualize the style and semantic attention of two examples in Fig. 5. Fig. 5 indicates that the context style attention focuses on general information from prototypes and template, respectively, while the semantic attention focuses on specific information.

Table 2: Human evaluation results. SBS: Stochastic Beam Search. HR: Human Responses.

<table>
<thead>
<tr>
<th>Metric</th>
<th>P2-Net vs SBS</th>
<th>P2-Net vs HR</th>
<th>SBS vs HR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wins</td>
<td>93</td>
<td>100</td>
<td>104</td>
</tr>
<tr>
<td>Ties</td>
<td>54</td>
<td>110</td>
<td>97</td>
</tr>
<tr>
<td>Losses</td>
<td>103</td>
<td>98</td>
<td>44</td>
</tr>
<tr>
<td>Wins</td>
<td>87</td>
<td>94</td>
<td>108</td>
</tr>
<tr>
<td>Ties</td>
<td>51</td>
<td>43</td>
<td>109</td>
</tr>
<tr>
<td>Losses</td>
<td>112</td>
<td>98</td>
<td>47</td>
</tr>
<tr>
<td>Wins</td>
<td>96</td>
<td>94</td>
<td>108</td>
</tr>
<tr>
<td>Ties</td>
<td>43</td>
<td>50</td>
<td>106</td>
</tr>
<tr>
<td>Losses</td>
<td>111</td>
<td>108</td>
<td>50</td>
</tr>
</tbody>
</table>

5.4 Qualitative analysis

For a qualitative understanding of the different responses from P2-Net and stochastic beam search, we list some examples in Table 3. From the good case, we see that although the responses from both models look good in terms of semantics, the responses from P2-Net are more diverse in terms of speaking styles. E.g., P2-Net
Table 3: Qualitative analysis with case studies.

<table>
<thead>
<tr>
<th>Good case</th>
<th>Diverse generations from P2-Net</th>
<th>Diverse generations from stochastic beam search</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) Would you like to try La Mimosa or Shiraz?</td>
<td>(1) You can choose from La Mimosa, Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(2) The La Mimosa and Shiraz are both available</td>
<td>(2) You can choose from La Mimosa or Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(3) What about La Mimosa or Shiraz?</td>
<td>(3) You can choose from La Mimosa or Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(4) I can recommend the La Mimosa or Shiraz.</td>
<td>(4) Sure, we have the La Mimosa or Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(5) Okay. Would you like to try La Mimosa or Shiraz?</td>
<td>(5) You can choose from La Mimosa or Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(6) How about the La Mimosa or Shiraz?</td>
<td>(6) Sure, we have the La Mimosa and the Shiraz.</td>
<td></td>
</tr>
<tr>
<td>(7) I have the La Mimosa, Shiraz.</td>
<td>(7) Sure, we have the La Mimosa and the Shiraz. Do you want to book a table for you?</td>
<td></td>
</tr>
<tr>
<td>(8) Okay, I have two options for you. La Mimosa and Shiraz.</td>
<td>(8) Sure, we have La Mimosa and Shiraz. Do you want to go?</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bad case</th>
<th>Diverse generations from P2-Net</th>
<th>Diverse generations from stochastic beam search</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) I have several trains. One arrives at 16:07 and the other at 24:07.</td>
<td>(1) There are several trains that fit your criteria. One arrives at 16:07 and the other at 24:07.</td>
<td></td>
</tr>
<tr>
<td>(2) There are several trains, arriving by 16:07 or arriving at 24:07.</td>
<td>(2) There are several trains that fit your criteria. One arrives at 16:07 and the latest at 24:07.</td>
<td></td>
</tr>
<tr>
<td>(3) There are several trains that would get you there at 16:07, or would you like to take one at 24:07?</td>
<td>(3) There are several trains that fit your criteria. One will get you there by 16:07 and 24:07.</td>
<td></td>
</tr>
<tr>
<td>(4) I have several trains that arrive by 16:07 and 24:07.</td>
<td>(4) There are several trains that fit your criteria. One will get you there by 16:07 and the other arrives at 24:07.</td>
<td></td>
</tr>
</tbody>
</table>

and stochastic beam search use different sentence patterns such as statements and questions, but P2-Net will generate different styles for statements, e.g., ‘... are both available’, ‘I can recommend ...’, ‘Okay, I have two options for you ...’, and for questions, e.g., ‘Would you like to try ...?’, ‘What about ...?’, ‘How about ...?’ The responses from stochastic beam search are less diverse. Most responses are statements, and their speaking styles do not change much, e.g., ‘You can choose from ...’ occurs 4 times.

For the bad case in Table 3, we see that: (i) The generated responses are not always precise or consistent in terms of semantics, e.g., in response (1) of P2-Net, there are ‘several’ trains in the first sentence, however, it generates ‘One ... and the other ...’ in the second sentence. This happens for all 4 responses from the stochastic beam search. (ii) The models do not take the template into account as much as expected. And when generating the responses, both models regard the two slot values as the only options, which clearly ignores some semantics in the template.

In both types of examples, stochastic beam search almost always puts the slots at the same position. The start is often the same because the beams are biased towards selecting slots early. During generation, the non-slot words from beam search often have a probability of less than 10% due to the large vocabulary. In contrast, slots tend to have a probability close to 100% because of the small set of slots. Thus, beams having slots early in the output have a significantly higher probability. Sampling prototypes in P2-Net does not suffer from this issue: we are not comparing different outputs on probabilities, but just sampling input styles.

6 CONCLUSION AND FUTURE WORK

Motivated by the finding that two key dimensions determine overall user satisfaction with TDSs: utility and user experience, we combine the merits of template-based dialogue response generation (DRG) and corpus-based dialogue response generation (DRG) in task-oriented dialogue systems (TDSs) in P2-Net, which is based on prototype guided paraphrasing. P2-Net can learn to extract style information from prototypes and extract semantics from template responses. By combining both during generating, P2-Net can generate more diverse responses (to improve the user experience) while preserving the semantics of template responses (to maintain utility). Automatic and human evaluations as well as a qualitative analysis demonstrate the effectiveness of P2-Net in terms of generating more diverse and human-like responses.

A limitation of P2-Net is that, in some cases, it will generate inconsistent content in the response and neglect some semantics in the template responses, which is not reflected by the slots. As to future work, on the one hand, we hope to incorporate mechanisms to address those issues [47]. On the other hand, we want to study how to apply P2-Net to other domains and languages with minimum effort in creating new datasets using transfer learning [51] or meta learning techniques [25, 40]. Finally, we would like to extend P2-Net to modern Transformer-based architectures [44], leveraging their recent success in many NLP domains [3, 6, 31, 32, 44].

Code and data. The dataset and code used to produce the results in this paper are shared at: https://github.com/phlippe/P2_Net.
Acknowledgements. This research was partially supported by Ahoel Delhaize and by the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl. All content represents the opinions of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES


