ScriptWriter: Narrative-Guided Script Generation

Yutao Zhu[1], **Ruihua Song**[2,*], **Zhicheng Dou**[3,*], **Jian-Yun Nie**[1], **Jin Zhou**[4]
[1]Université de Montréal, Montréal, Québec, Canada
[2]Microsoft XiaoIce, Beijing, China
[3]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[4]Beijing Film Academy, Beijing, China
yutao.zhu@umontreal.ca, rsong@microsoft.com
dou@ruc.edu.cn, nie@iro.umontreal.ca, whitezj@vip.sina.com

Abstract

It is appealing to have a system that generates a story or scripts automatically from a storyline, even though this is still out of our reach. In dialogue systems, it would also be useful to drive dialogues by a dialogue plan. In this paper, we address a key problem involved in these applications - guiding a dialogue by a narrative. The proposed model ScriptWriter selects the best response among the candidates that fit the context as well as the given narrative. It keeps track of what in the narrative has been said and what is to be said. A narrative plays a different role than the context (i.e., previous utterances), which is generally used in current dialogue systems. Due to the unavailability of data for this new application, we construct a new large-scale data collection *GraphMovie* from a movie website where end-users can upload their narratives freely when watching a movie. Experimental results on the dataset show that our proposed approach based on narratives significantly outperforms the baselines that simply use the narrative as a kind of context.

1 Introduction

Narrative is generally understood as a way to tell a story. WordNet defines it as "a message that tells the particulars of an act or occurrence or course of events; presented in writing or drama or cinema or as a radio or television program"[1]. Narrative plays an important role in many natural language processing (NLP) tasks. For example, in storytelling, the storyline is a type of narrative, which helps generate coherent and consistent stories (Fan et al., 2018, 2019). In dialogue generation, narrative can be used to define a global plan for the whole conversation session, so as to avoid generating inconsistent and scattered responses (Xing et al., 2018; Tian et al., 2017; Ghazvininejad et al., 2018).

Narrative	Jenny **doesn't like to go home**. To accompany Jenny, Gump decides to **go home later**. Gump is Jenny's **best friend**. 珍妮不喜欢回家。为了陪珍妮，甘决定晚点回家。甘是珍妮最好的朋友。	
Initial line	Mama's going to worry about me. 妈会担心我的	
1st line	Just **stay a little longer**. 再坐一会!	✓
	Yeah, and I'll bet you $ 10,000 he laughs his ass off. 我打赌他会笑破肚皮	
2nd line	Ok, Jenny, **I'll stay**. 好，珍妮，我留下来	✓
	She lived in an old house. 她家的房子破旧	
3rd line	He was a very loving man. 他是个非常有爱心的人	
	You are my most **special friend**. 你是我最特别的朋友	✓

Figure 1: An example of part of a script with a narrative extracted from our *GraphMovie* dataset. The checked lines are from a ground-truth session, while the unchecked responses are other candidates that are relevant but not coherent with the narrative.

In this work, we investigate the utilization of narratives in a special case of text generation – **movie script generation**. This special form of conversation generation is chosen due to the unavailability of the data for a more general form of application. Yet it does require the same care to leverage narratives in general conversation, and hence can provide useful insight to a more general form of narrative-guided conversation. The dataset we use to support our study is collected from GraphMovie[2], where an end-user retells the story of a movie by uploading descriptive paragraphs in his/her own words. More details about the dataset will be presented in Section 3.2. An example is shown in Figure 1, where the narrative

[*] Corresponding authors.

[1] http://wordnetweb.princeton.edu/perl/webwn?s=narrative

[2] http://www.graphmovies.com/home/2/index.php. Unfortunately, we find this website was closed recently.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8647–8657
July 5 - 10, 2020. ©2020 Association for Computational Linguistics

is uploaded to retell several lines of a script in a movie. Our task is to generate/select the following lines by leveraging the narrative.

Our problem is closely related to dialogue generation that takes into account the context (Wu et al., 2017; Zhang et al., 2018; Zhou et al., 2018b). However, a narrative plays a different and more specific role than a general context. In particular, a narrative may cover the whole story (a part of a script), thus a good conversation should also cover all the aspects mentioned in a narrative, which is not required with a general context. In this paper, we propose a new model called **ScriptWriter** to address the problem of script generation/selection with the help of a narrative. ScriptWriter keeps track of what in the narrative has been said and what is remaining to select the next line by an updating mechanism. The matching between updated narrative, context, and response are then computed respectively and finally aggregated as a matching score. As it is difficult to evaluate the quality of script generation, we frame our work in a more restricted case - selecting the right response among a set of candidates. This form of more limited conversation generation - retrieval-based conversation - has been widely used in the previous studies (Wu et al., 2017; Zhou et al., 2018b), and it provides an easier way to evaluate the impact of narratives.

We conduct experiments on a dataset we collected and made publicly available (see Section 5). The experiments will show that using a narrative to guide the generation/selection of script is a much more appropriate approach than using it as part of the general context.

Our work has three main contributions:

(1) To our best knowledge, this is the first investigation on movie script generation with a narrative. This task could be further extended to a more general text generation scenario when suitable data are available.

(2) We construct the first large-scale data collection *GraphMovie* to support research on narrative-guided movie script generation, which is made publicly accessible.

(3) We propose a new model in which a narrative plays a specific role in guiding script generation. This will be shown to be more appropriate than a general context-based approach.

2 Related Work

2.1 Narrative Understanding

It has been more than thirty years since researchers proposed "narrative comprehension" as an important ability of artificial intelligence (Rapaport et al., 1989). The ultimate goal is the development of a computational theory to model how humans understand narrative texts. Early explorations used symbolic methods to represent the narrative (Turner, 1994; Bringsjord and Ferrucci, 1999) or rule-based approaches to generate the narrative (Riedl and Young, 2010). Recently, deep neural networks have been used to tackle the problem (Bamman et al., 2019), and related problems such as generating coherent and cohesive text (Cho et al., 2019) and identifying relations in generated stories (Roemmele, 2019) have also been addressed. However, these studies only focused on how to understand a narrative itself (e.g., how to extract information from a narrative). They did not investigate how to utilize the narrative in an application task such as dialogue generation.

2.2 Dialogue Systems

Existing methods of open-domain dialogue can be categorized into two groups: retrieval-based and generation-based. Recent work on response generation is mainly based on sequence-to-sequence structure with attention mechanism (Shang et al., 2015; Vinyals and Le, 2015), with multiple extensions (Li et al., 2016; Xing et al., 2017; Zhou et al., 2018a, 2020; Zhu et al., 2020). Retrieval-based methods try to find the most reasonable response from a large repository of conversational data, instead of generating a new one (Wu et al., 2017; Zhou et al., 2018b; Zhang et al., 2018). In general, the utterances in the previous turns are taken together as the context for selecting the next response. Retrieval-based methods are widely used in real conversation products due to their more fluent and diverse responses and better efficiency. In this paper, we focus on extending retrieval-based methods by using a narrative as a plan for a session. This is a new problem that has not been studied before.

Contrary to open-domain chatbots, task-oriented systems are designed to accomplish tasks in a specific domain (Seneff et al., 1998; Levin et al., 2000; Wang et al., 2011; Tur and Mori, 2011). In these systems, a dialogue state tracking component is designed for tracking what has happened in a dia-

Table 1: Statistics of *GraphMovie* corpus.

	Training	Validation	Test
# Sessions	14,498	805	806
# Micro-sessions	136,524	37,480	38,320
# Candidates	2	10	10
Min. #turns	2	2	2
Max. #turns	34	27	17
Avg. #turns	4.71	4.66	4.75
Avg. #words in Narr.	25.04	24.86	24.18

logue (Williams and Young, 2007; Henderson et al., 2014; Xu and Rudnicky, 2000). This inspires us to track the remaining information in the narrative that has not been expressed by previous lines of conversation. However, existing methods cannot be applied to our task directly as they are usually predefined for specific tasks, and the state tracking is often framed as a classification problem.

2.3 Story Generation

Existing studies have also tried to generate a story. Early work relied on symbolic planning (Meehan, 1977; Cavazza et al., 2002) and case-based reasoning (y Pérez and Sharples, 2001; Gervás et al., 2005), while more recent work uses deep learning methods. Some of them focused on story ending generation (Peng et al., 2018; Guan et al., 2019), where the story context is given, and the model is asked to select a coherent and consistent story ending. This is similar to the dialogue generation problem mentioned above. Besides, attempts have been made to generate a whole story from scratch (Fan et al., 2018, 2019). Compared with the former task, this latter is more challenging since the story framework and storyline should all be controlled by the model.

Some recent studies also tried to guide the generation of dialogues (Wu et al., 2019; Tang et al., 2019) or stories (Yao et al., 2019) with keywords - the next response is asked to include the keywords. This is a step towards guided response generation and bears some similarities with our study. However, a narrative is more general than keywords, and it provides a description of the dialogue session rather than imposing keywords to the next response.

3 Problem Formulation and Dataset

3.1 Problem Formulation

Suppose that we have a dataset $\mathcal{D}$, in which a sample is represented as (y, c, p, r), where $c = \{s_1, \cdots, s_n\}$ represents a context formed by the preceding sentences/lines $\{s_i\}_{i=1}^n$; p is a predefined narrative that governs the whole script session, and r is a next line candidate (we refer to it as a response); $y \in \{0, 1\}$ is a binary label, indicating whether r is a proper response for the given c and p. Intuitively, a proper response should be relevant to the context, and be coherent and aligned with the narrative. Our goal is to learn a model $g(c, p, r)$ with $\mathcal{D}$ to determine how suitable a response r is to the given context c and narrative p.

3.2 Data Collection and Construction

Data is a critical issue in research on story/dialogue generation. Unfortunately, no dataset has been created for narrative-guided story/dialogue generation. To fill the gap, we constructed a test collection from GraphMovie, where an editor or a user can retell the story of a movie by uploading descriptive paragraphs in his/her own words to describe screenshots selected from the movie. A movie on this website has, on average, 367 descriptions. A description paragraph often contains one to three sentences to summarize a fragment of a movie. It can be at different levels - from retelling the same conversations to a high-level description. We consider these descriptions as narratives for a sequence of dialogues, which we call a session in this paper. Each dialogue in a session is called a line of script (or simply a line).

To construct the dataset, we use the top 100 movies in IMDB[3] as an initial list. For each movie, we collect its description paragraphs from GraphMovie. Then we hire annotators to watch the movie and annotate the start time and end time of the dialogues corresponding to each description paragraph through an annotation tool specifically developed for this purpose. According to the start and end time, the sequence of lines is extracted from the subtitle file and aligned with a corresponding description paragraph.

As viewers of a movie can upload descriptions freely, not all description paragraphs correspond to a narrative and are suitable for our task. For example, some uploaded paragraphs express one's subjective opinions about the movie, the actors, or simply copy the script. Therefore, we manually review the data and remove such non-narrative data. We also remove sessions that have less than two lines. Finally, we obtain 16,109 script sessions,

[3] https://www.imdb.com/

each of which contains a description paragraph (narrative) and corresponding lines of the script. As shown in Table 1, on average, a narrative has about 25 words, and a session has 4.7 lines. The maximum number of lines in a session is 34.

Our task is to select one response from a set of candidates at any point during the session. By moving the prediction point through the session, we obtain a set of micro-sessions, each of which has a sequence of previous lines as context at that point of time, the same narrative as the session, and the next line to predict. The candidates to be selected contain one ground-truth line - the one that is genuinely the next line, together with one (in the training set) or nine (in the validation/test set) other candidates retrieved with the previous lines by Solr[4]. The above preparation of the dataset follows the practice in the literature (Wu et al., 2017) for retrieval-based dialogue.

4 Proposed Method: ScriptWriter

4.1 Overview

A good response is required to be coherent with the previous lines, i.e., context, and be consistent with the given narrative. For example, “Just stay a little longer” can respond “Mama’s going to worry about me” and it has no conflict with the narrative in Figure 1. Furthermore, as our target is to generate all lines in the session successively, it is also required that the following lines should convey the information that the former lines have not conveyed. Otherwise, only a part of the narrative is covered, and we will miss some other aspects specified in the narrative.

We propose an attention-based model called ScriptWriter to solve the problem. ScriptWriter follows a representation-matching-aggregation framework. First, the narrative, the context, and the response candidate are represented in multiple granularities by multi-level attentive blocks. Second, we propose an updating mechanism to keep track of what in a narrative has been expressed and explicitly lower their weights in the updated narrative so that more emphasis can be put on the remaining parts. Third, matching features are extracted between different elements: between context and response to capture whether it is a proper reply; between narrative and response to capture whether it is consistent with the narrative; and between context and narrative to implicitly track what in the narrative has been expressed in the previous lines. Finally, the above matching features are concatenated together and a final matching score is produced by convolutional neural networks (CNNs) and a multi-layer perceptron (MLP).

4.2 Representation

To better handle the gap in words between two word sequences, we propose to use an attentive block, which is similar to that used in Transformer (Vaswani et al., 2017). The input of an attentive block consists of three sequences, namely query ($\mathbf{Q}$), key ($\mathbf{K}$), and value ($\mathbf{V}$). The output is a new representation of the query and is denoted as AttentiveBlock($\mathbf{Q}, \mathbf{K}, \mathbf{V}$) in the remaining parts. This structure is used to represent a response, lines in the context, and a narrative.

More specifically, given a narrative $p = (w_1^p, \cdots, w_{n_p}^p)$, a line $s_i = (w_1^{s_i}, \cdots, w_{n_{s_i}}^{s_i})$ and a response candidate $r = (w_1^r, \cdots, w_{n_r}^r)$, ScriptWriter first uses a pre-trained embedding table to map each word w to a d_e-dimension embedding $\mathbf{e}$, i.e., $w \Rightarrow \mathbf{e}$. Thus the narrative p, the line s_i and the response candidate r are represented by matrices $\mathbf{P}^0 = (\mathbf{e}_1^p, \cdots, \mathbf{e}_{n_p}^p)$, $\mathbf{S}_i^0 = (\mathbf{e}_1^{s_i}, \cdots, \mathbf{e}_{n_{s_i}}^{s_i})$ and $\mathbf{R}^0 = (\mathbf{e}_1^r, \cdots, \mathbf{e}_{n_r}^r)$.

Then ScriptWriter takes $\mathbf{P}^0$, $\{\mathbf{S}_i^0\}_{i=1}^n$ and $\mathbf{R}^0$ as inputs and uses stacked attentive blocks to construct multi-level self-attention representations. The output of the $(l-1)^{th}$ level of attentive block is input into the l^{th} level. The representations of p, s_i, and r at the l^{th} level are defined as follows:

$$\mathbf{P}^l = \text{AttentiveBlock}(\mathbf{P}^{l-1}, \mathbf{P}^{l-1}, \mathbf{P}^{l-1}), \quad (1)$$

$$\mathbf{S}_i^l = \text{AttentiveBlock}(\mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}), \quad (2)$$

$$\mathbf{R}^l = \text{AttentiveBlock}(\mathbf{R}^{l-1}, \mathbf{R}^{l-1}, \mathbf{R}^{l-1}), \quad (3)$$

where l ranges from 1 to L.

Inspired by a previous study (Zhou et al., 2018b), we apply another group of attentive blocks, which is referred to as cross-attention, to capture semantic dependency between p, s_i and r. Considering p and s_i at first, their cross-attention representations are defined by:

$$\overline{\mathbf{P}}_{s_i}^l = \text{AttentiveBlock}(\mathbf{P}^{l-1}, \mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}), \quad (4)$$

$$\overline{\mathbf{S}}_{i,p}^l = \text{AttentiveBlock}(\mathbf{S}_i^{l-1}, \mathbf{P}^{l-1}, \mathbf{P}^{l-1}). \quad (5)$$

Here, the words in the narrative can attend to all words in the line, and vice verse. In this way, some inter-dependent segment pairs, such as “stay” in the

[4] https://lucene.apache.org/solr/

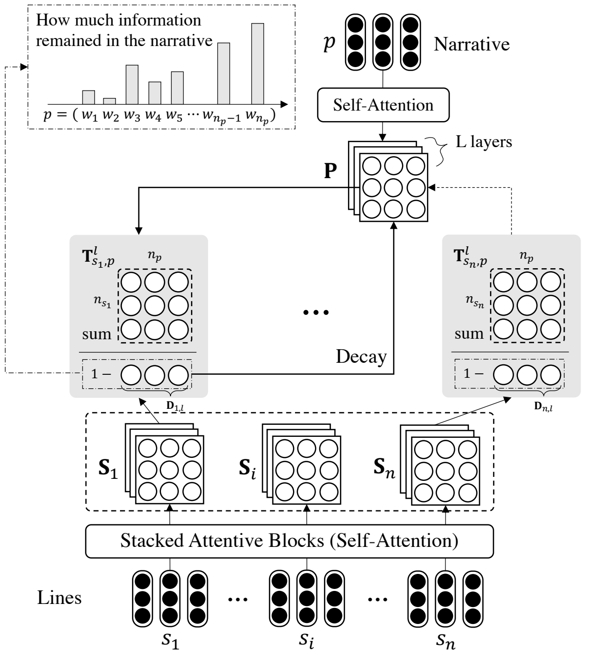

Figure 2: Updating mechanism in ScriptWriter. The representation of the narrative is updated by lines in the context one by one. The information that has been expressed is decayed. Thus the updated narrative focuses more on the remaining information.

line and "go home later" in the narrative, become close to each other in the representations. Similarly, we compute cross-attention representations between p and r and between r and s_i at different levels, which are denoted as $\overline{\mathbf{P}}^l_r$, $\overline{\mathbf{R}}^l_p$, $\overline{\mathbf{S}}^l_{i,r}$ and $\overline{\mathbf{R}}^l_{s_i}$. These representations further provide matching information across different elements in the next step.

4.3 Updating Mechanism

We design an updating mechanism to keep track of the coverage of the narrative by the lines so that the selection of the response will focus on the uncovered parts. The mechanism is illustrated in Figure 2. We update a narrative gradually by all lines in the context one by one. For the i^{th} line s_i, we conduct a matching between $\mathbf{S}_i$ and $\mathbf{P}$ by their cosine similarity at all levels (l) of attentive blocks:

$$\mathbf{T}^l_{s_i,p}[j][k] = \cos(\mathbf{S}^l_i[j], \mathbf{P}^l[k]), \tag{6}$$

where j and k stand for the j^{th} word in s_i and k^{th} word in p respectively. To summarize how much information in p has been expressed by s_i, we compute a vector $\mathbf{D}_i$ by conducting summations along vertical axis on each level in the matching map $\mathbf{T}_{s_i,p}$. The summation on the l^{th} level is:

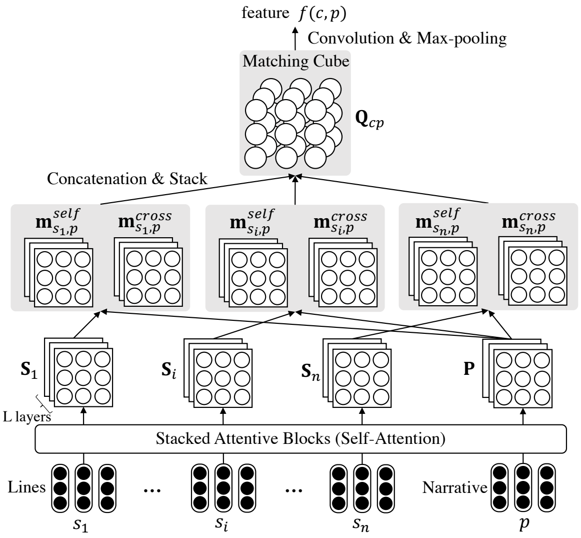
feature $f(c,p)$
Convolution & Max-pooling
Matching Cube
$\mathbf{Q}_{cp}$
Concatenation & Stack
$\mathbf{m}^{self}_{s_1,p}$
$\mathbf{m}^{cross}_{s_1,p}$
$\mathbf{m}^{self}_{s_i,p}$
$\mathbf{m}^{cross}_{s_i,p}$
$\mathbf{m}^{self}_{s_n,p}$
$\mathbf{m}^{cross}_{s_n,p}$
$\mathbf{S}_1$
$\mathbf{S}_i$
$\mathbf{S}_n$
$\mathbf{P}$
L layers
Stacked Attentive Blocks (Self-Attention)
Lines
...
Narrative
s_1
s_i
s_n
p

Figure 3: The context-narrative matching. All lines and the narrative are represented by attentive blocks and the matching between them results in a matching cube $\mathbf{Q}_{cp}$. Matching features are aggregated and distilled by a CNN.

$$\mathbf{D}^l_i = [d^l_{i,1}, d^l_{i,2}, \cdots, d^l_{i,n_p}], \tag{7}$$

$$d^l_{i,k} = \gamma \sum_{j=1}^{n_{s_i}} \mathbf{T}^l_{s_i,p}[j][k], \tag{8}$$

where n_p, n_{s_i} denotes the number of words in p and s_i; $\gamma \in [0, 1]$ is a parameter to learn and works as a gate to control the decaying degree of the mentioned information. Finally, we update the narrative's representation as follows for the i^{th} line s_i in the context:

$$\mathbf{P}^l_{i+1} = (1 - \mathbf{D}^l_i)\mathbf{P}^l_i. \tag{9}$$

The initial representation $\mathbf{P}^l_0$ is equal to $\mathbf{P}^l$ defined in Equation (1). If there are n lines in the context, this update is executed n times, and $(1 - \mathbf{D}^l)$ will produce a continuous decaying effect.

4.4 Matching

The matching between the narrative p and the line s_i is conducted based on both their self-attention and cross-attention representations, as shown in Figure 3.

First, ScriptWriter computes the dot product on these two representations separately as follows:

$$\mathbf{m}^{self}_{s_i,p,l}[j,k] = \mathbf{S}^l_i[j]^T \cdot \mathbf{P}^l[k], \tag{10}$$

$$\mathbf{m}^{cross}_{s_i,p,l}[j,k] = \overline{\mathbf{S}}^l_{i,p}[j]^T \cdot \overline{\mathbf{P}}^l_{s_i}[k], \tag{11}$$

where l ranges from 0 to L. Each element is the dot product of the j^{th} word representation in $\mathbf{S}_i^l$ or $\overline{\mathbf{S}}_{i,p}^l$ and the k^{th} word representation in $\mathbf{P}^l$ or $\overline{\mathbf{P}}_{s_i}^l$. Then the matching maps in different layers are concatenated together as follows:

$$\mathbf{m}_{s_i,p}^{self}[j,k] = \left[\mathbf{m}_{s_i,p,0}^{self}[j,k]; \cdots ; \mathbf{m}_{s_i,p,L}^{self}[j,k]\right],$$
$$\mathbf{m}_{s_i,p}^{cross}[j,k] = \left[\mathbf{m}_{s_i,p,0}^{cross}[j,k]; \cdots ; \mathbf{m}_{s_i,p,L}^{cross}[j,k]\right],$$

where $[;]$ is concatenation operation. Finally, the matching features computed by the self-attention representation and the cross-attention representation are fused as follows:

$$\mathbf{M}_{s_i,p}[j,k] = \left[\mathbf{m}_{s_i,p}^{self}[j,k]; \mathbf{m}_{s_i,p}^{cross}[j,k]\right].$$

The matching matrices $\mathbf{M}_{p,r}$ and $\mathbf{M}_{s_i,r}$ for narrative-response and context-response are constructed in a similar way. For the sake of brevity, we omit the formulas. After concatenation, each cell in $\mathbf{M}_{s_i,p}$, $\mathbf{M}_{p,r}$ or $\mathbf{M}_{s_i,r}$ has $2(L+1)$ channels and contains matching information at different levels.

The matching between narrative, context, and response serves for different purposes. Context-response matching ($\mathbf{M}_{s_i,r}$) serves to select a response suitable for the context. Context-narrative matching ($\mathbf{M}_{s_i,p}$) helps the model "remember" how much information has been expressed and implicitly influences the selection of the next responses. Narrative-response matching ($\mathbf{M}_{p,r}$) helps the model to select a more consistent response with the narrative. As the narrative keeps being updated along with the lines in context, ScriptWriter tends to dynamically choose the response that matches what remains unexpressed in the narrative.

4.5 Aggregation

To further use the information across two consecutive lines, ScriptWriter piles up all the context-narrative matching matrices and all the context-response matching matrices to construct two cubes $\mathbf{Q}_{cp} = \{\mathbf{M}_{s_i,p}[j,k]\}_{i=1}^n$ and $\mathbf{Q}_{cr} = \{\mathbf{M}_{s_i,r}[j,k]\}_{i=1}^n$, where n is the number of lines in the session. Then ScriptWriter employs 3D convolutions to distill important matching features from the whole cube. We denote these two feature vectors as $f(c,p)$ and $f(c,r)$. For narrative-response matching, ScriptWriter conducts 2D convolutions on $\mathbf{M}_{p,r}$ to distill matching features between the narrative and the response, denoted as $f(p,r)$.

The three types of matching features are concatenated together, and the matching score $g(c,p,r)$ for ranking response candidates is computed by an MLP with a sigmoid activation function, which is defined as:

$$f(c,p,r) = [f(c,p); f(c,r); f(p,r)], \quad (12)$$
$$g(c,p,r) = \text{sigmoid}(\mathbf{W}^T f(c,p,r) + b), \quad (13)$$

where $\mathbf{W}$ and b are parameters.

ScriptWriter learns $g(c,p,r)$ by minimizing cross entropy with $\mathcal{D}$. The objective function is formulated as:

$$L(\theta) = -\sum_{(y,c,p,r)\in\mathcal{D}} [y \log(g(c,p,r)) + (1-y)\log(1-g(c,p,r))]. \quad (14)$$

5 Experiments

5.1 Evaluation setup

As presented in Table 1, we randomly split the the *GraphMovie* collection into training, validation and test set. The split ratio is 18:1:1. We split the sessions into micro-sessions: given a session with n lines in the context, we will split it into n micro-sessions with length varying from 1 to n. These micro-sessions share the same narrative. By doing this, the model is asked to learn to select one line as the response from a set of candidates at any point during the session, and the dataset, in particular for training, can be significantly enlarged.

We conduct two kinds of evaluation as follows:

Turn-level task asks a model to rank a list of candidate responses based on its given context and narrative for a micro-session. The model then selects the best response for the current turn. This setting is similar to the widely studied response selection task (Wu et al., 2017; Zhou et al., 2018b; Zhang et al., 2018). We follow these previous studies and employ recall at position k in n candidates ($\mathrm{R_n@k}$) and mean reciprocal rank (MRR) (Voorhees, 1999) as evaluation metrics. For example, $\mathrm{R_{10}@1}$ means recall at one when we rank ten candidates (one positive sample and nine negative samples). The final results are average numbers over all micro-sessions in the test set.

Session-level task aims to predict all the lines in a session gradually. It starts with the first line of the session as the context and the given narrative and predicts the best next line. The predicted line is then incorporated into the context to predict the next line. This process continues until the last line of the session is selected. Finally, we calculate precision over the whole original session and report average numbers over all sessions in the test set. Precision is defined as the number of correct selection divided by the number of lines in a session. We consider two measures: 1) $\mathrm{P}_{\mathrm{strict}}$ which accepts a right response at the right position; 2) $\mathrm{P}_{\mathrm{weak}}$ which accepts a right response at any position.

5.2 Baselines

As no previous work has been done on narrative-based script generation, no proper baseline exists. Nevertheless, some existing multi-turn conversation models based on context can be adapted to work with a narrative: the context is simply extended with the narrative. Two different extension methods have been tested: the narrative is added into the context together with the previous lines; the narrative is used as a second context. In the latter case, two matching scores are obtained for context-narrative and narrative-response. They are aggregated through an MLP to produce a final score. This second approach turns out to perform better. Therefore, we only report the results with this latter method[5].

(1) MVLSTM (Wan et al., 2016): it concatenates all previous lines as a context and uses an LSTM to encode the context and the response candidate. A matching score is determined by an MLP based on a map of cosine similarity between them. A matching score for narrative-response is produced similarly.

(2) DL2R (Yan et al., 2016): it encodes the context by an RNN followed by a CNN. The matching score is computed similarly to MVLSTM.

(3) SMN (Wu et al., 2017): it matches each line with response sequentially to produce a matching vector with CNNs. The matching vectors are aggregated with an RNN.

(4) DAM (Zhou et al., 2018b): it represents a context and a response by using self-attention and cross-attention operation on them. It uses CNNs to extract features and uses an MLP to get a score. Different from our model, this model only considers the context-response matching and does not track what in the narrative has already been expressed by the previous lines, i.e., context.

(5) DUA (Zhang et al., 2018): it concatenates the last line with each previous line in the context and response, respectively. Then it performs a self-attention operation to get refined representations, based on which matching features are extracted with CNNs and RNNs.

5.3 Training Details

All models are implemented in Tensorflow[6]. Word embeddings are pre-trained by Word2vec (Mikolov et al., 2013) on the training set with 200 dimensions. We test the stack number in $\{1,2,3\}$ and report our results with three stacks. Due to the limited resources, we cannot conduct experiments with a larger number of stacks, which could be tested in the future. Two 3D convolutional layers have 32 and 16 filters, respectively. They both use [3,3,3] as kernel size, and the max-pooling size is [3,3,3]. Two 2D convolutional layers on narrative-response matching have 32 and 16 filters with [3,3] as kernel size. The max-pooling size is also [3,3]. All parameters are optimized with Adam optimizer (Kingma and Ba, 2015). The learning rate is 0.001 and decreased during training. The initial value for γ is 0.5. The batch size is 64. We use the validation set to select the best models and report their performance on the test set. The maximum number of lines in context is set as ten, and the maximum length of a line, response, and narrative sentence is all set as 50. All sentences are zero-padded to the maximum length. We also padded zeros if the number of lines in a context is less than 10. Otherwise, we kept the latest ten lines. The dataset and the source code of our model are available on GitHub[7].

5.4 Results and Analysis

5.4.1 Evaluation Results

The experimental results are reported in Table 2. The results on both turn-level and session-level evaluations indicate that ScriptWriter dramatically outperforms all baselines, including DAM and DUA, which are two state-of-the-art models on multi-turn response selection. All improvements are statistically significant (p-value ≤ 0.01). DAM performs better than other baselines, which confirms the effectiveness of the self and cross attention mechanism used in this model. The DUA model also uses the attention mechanism. It outper-

[5] We also tested some basic models such as RNN, LSTM, and BiLSTM (Lowe et al., 2015) in our experiments. However, they cannot achieve comparable results to the selected baselines.

[6] https://www.tensorflow.org

[7] https://github.com/DaoD/ScriptWriter

Table 2: Evaluation results on two response selection tasks: turn-level and session-level. Our ScriptWriter model is represented as SW. † and ⋆ denote significant improvements with SW in t-test with $p \leq 0.01$ and $p \leq 0.05$ respectively.

	Turn-level				Session-level	
Method	R_2@1	R_{10}@1	R_{10}@5	MRR	P_{strict}	P_{weak}
MVLSTM	0.651†	0.217†	0.732†	0.395†	0.198†	0.224†
DL2R	0.643†	0.210†	0.638†	0.314†	0.230†	0.243†
SMN	0.641†	0.176†	0.696†	0.392†	0.197†	0.236†
DAM	0.631†	0.240†	0.733†	0.408†	0.226†	0.236†
DUA	0.654†	0.237†	0.736†	0.396†	0.223†	0.251†
SW	**0.730**	**0.365**	**0.814**	**0.503**	**0.373**	**0.383**
$\text{SW}_{\text{static}}$	0.723	0.351	0.801	0.484†	0.338†	0.366
SW-PR	0.654†	0.246†	0.721†	0.398†	0.223†	0.239†
SW-CP	0.710⋆	0.326†	0.793†	0.473†	0.329†	0.352†
SW-CR	0.725	0.316†	0.766†	0.466†	0.335†	0.382

forms the other baselines that do not use attention. Both observations confirm the advantage of using attention mechanisms over pure RNN.

Between the two session-level measures, we observe that our model is less affected when moving from P_{weak} to P_{strict}. This shows that ScriptWriter can better select a response in the right position. We attribute this behavior to the utilization of narrative coverage.

5.4.2 Model Ablation

We conduct an ablation study to investigate the impact of different modules in ScriptWriter. First, we remove the updating mechanism by setting $\gamma = 0$ (i.e., the representation of the narrative is not updated but static). This model is denoted as $\text{ScriptWriter}_{\text{static}}$ in Table 2. Then we remove narrative-response, context-narrative, and matching-response, respectively. These variants are denoted as ScriptWriter-PR, ScriptWriter-CP, and ScriptWriter-CR.

Model ablation results are shown in the second part of Table 2. We have the following findings: 1) ScriptWriter performs better than $\text{ScriptWriter}_{\text{static}}$, demonstrating the effectiveness of updating mechanism for the narrative. The optimal value of γ is at around 0.647 after training, which means that only about 35% of information is kept when a line conveys it. 2) In both turn-level and session-level evaluations, the performance drops the most when we remove narrative-response matching. This indicates that the relevance of the response to the narrative is the most useful information in narrative-guided script generation. 3) When we remove context-narrative matching, the performance drops too, indicating that context-narrative matching may provide implicit and complementary information for controlling the alignment of response and narrative. 4) In contrast, when we remove the context-response matching, the performance also drops, however, at a much smaller scale, especially on P_{weak}, than when narrative-response matching is removed. This contrast indicates that narrative is a more useful piece of information than context to determine what should be said next, thus it should be taken into account with an adequate mechanism.

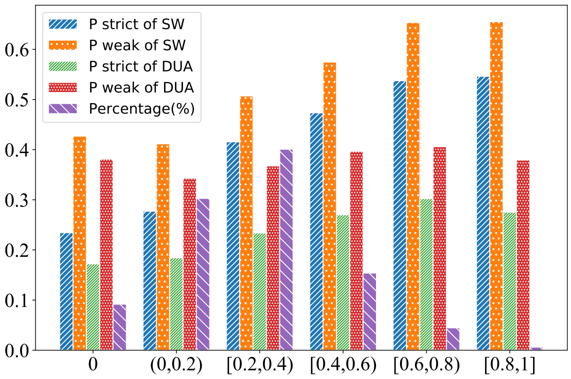

Figure 4: The performance of ScriptWriter (SW) and DUA on the test set with different types of narrative in session-level evaluation.

5.4.3 Performance across Narrative Types

As we explained, narratives in our dataset are contributed by netizens, and they vary in style. Some narratives are detailed, while others are general. The question we analyze is how general vs. detailed narratives affect the performance of response selection. We use a simple method to evaluate roughly the degree of detail of a narrative: a narrative that has a high lexical overlap with the lines in the session is considered to be detailed. Narratives are put into six buckets depending on their level of detail, as shown in Figure 4.

We plot the performance of ScriptWriter and DUA in session-level evaluation over different types of narratives. The first type "0" means no word overlap between narrative and dialogue sessions. This is the most challenging case, representing extremely general narratives. It is not surprising to see that both ScriptWriter and DUA performs poorly on this type compared with other types in terms of P_{strict}. The performance tends to become better when the overlap ratio is increased. This is consistent with our intuition: when a narrative is more detailed and better aligned with the session in wording, it is easier to choose the best responses. This plot also shows that our ScriptWriter can achieve better performance than DUA on all types of narratives, which further demonstrates the effectiveness of using narrative to guide the dialogue.

We also observe that the buckets "[0, 0.2)" and "[0.2, 0.4)" contain the largest proportions of narratives. This indicates that most netizens do not use the original lines to retell a story. The problem we address in this paper is thus non-trivial.

6 Conclusion and Future Work

Although story generation has been extensively studied in the literature, no existing work addressed the problem of generating movie scripts following a given storyline or narrative. In this paper, we addressed this problem in the context of generating dialogues in a movie script. We proposed a model that uses the narrative to guide the dialogue generation/retrieval. We keep track of what in the narrative has already been expressed and what is remaining to select the next line through an updating mechanism. The final selection of the next response is based on multiple matching criteria between context, narrative and response. We constructed a new large-scale data collection for narrative-guided script generation from movie scripts. This is the first public dataset available for testing narrative-guided dialogue generation/selection. Experimental results on the dataset showed that our proposed approach based on narrative significantly outperforms the baselines that use a narrative as an additional context, and showed the importance of using the narrative in a proper manner. As a first investigation on the problem, our study has several limitations. For example, we have not considered the order in the narrative description, which could be helpful in generating dialogues in correct order. Other methods to track the dialogue state and the coverage of narrative can also be designed. Further investigations are thus required to fully understand how narratives can be effectively used in dialogue generation.

Acknowledgments

Ruihua Song and Zhicheng Dou are the corresponding authors. This work was supported by National Natural Science Foundation of China No. 61872370 and No. 61832017, and Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098.

References

David Bamman, Snigdha Chaturvedi, Elizabeth Clark, Madalina Fiterau, and Mohit Iyyer, editors. 2019. *Proceedings of the First Workshop on Narrative Understanding*. Association for Computational Linguistics, Minneapolis, Minnesota.

Selmer Bringsjord and David Ferrucci. 1999. *Artificial intelligence and literary creativity: Inside the mind of brutus, a storytelling machine*. Psychology Press.

Marc Cavazza, Fred Charles, and Steven J. Mead. 2002. Planning characters' behaviour in interactive storytelling. *Journal of Visualization and Computer Animation*, 13(2):121–131.

Woon Sang Cho, Pengchuan Zhang, Yizhe Zhang, Xiujun Li, Michel Galley, Chris Brockett, Mengdi Wang, and Jianfeng Gao. 2019. Towards coherent and cohesive long-form text generation. In *Proceedings of the First Workshop on Narrative Understanding*, pages 1–11, Minneapolis, Minnesota. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. 2005. Story plot generation based on CBR. *Knowl. Based Syst.*, 18(4-5):235–242.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5110–5117.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of*

Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, pages 6473–6480.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Esther Levin, Shrikanth S. Narayanan, Roberto Pieraccini, Konstantin Biatov, Enrico Bocchieri, Giuseppe Di Fabbrizio, Wieland Eckert, Sungbok Lee, A. Pokrovsky, Mazin G. Rahim, P. Ruscitti, and Marilyn A. Walker. 2000. The at&t-darpa communicator mixed-initiative spoken dialog system. In *Sixth International Conference on Spoken Language Processing, ICSLP 2000 / INTERSPEECH 2000, Beijing, China, October 16-20, 2000*, pages 122–125.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

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, Prague, Czech Republic. Association for Computational Linguistics.

James R. Meehan. 1977. Tale-spin, an interactive program that writes stories. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence. Cambridge, MA, USA, August 22-25, 1977*, pages 91–98.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, New Orleans, Louisiana. Association for Computational Linguistics.

Rafael Pérez y Pérez and Mike Sharples. 2001. MEXICA: A computer model of a cognitive account of creative writing. *J. Exp. Theor. Artif. Intell.*, 13(2):119–139.

William J Rapaport, Erwin M Segal, Stuart C Shapiro, David A Zubin, Gail A Bruder, Judith Felson Duchan, and David M Mark. 1989. Cognitive and computer systems for understanding narrative text.

Mark O. Riedl and Robert Michael Young. 2010. Narrative planning: Balancing plot and character. *J. Artif. Intell. Res.*, 39:217–268.

Melissa Roemmele. 2019. Identifying sensible lexical relations in generated stories. In *Proceedings of the First Workshop on Narrative Understanding*, pages 44–52, Minneapolis, Minnesota. Association for Computational Linguistics.

Stephanie Seneff, Edward Hurley, Raymond Lau, Christine Pao, Philipp Schmid, and Victor Zue. 1998. GALAXY-II: a reference architecture for conversational system development. In *The 5th International Conference on Spoken Language Processing, Incorporating The 7th Australian International Speech Science and Technology Conference, Sydney Convention Centre, Sydney, Australia, 30th November - 4th December 1998*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586, Beijing, China. Association for Computational Linguistics.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634, Florence, Italy. Association for Computational Linguistics.

Zhiliang Tian, Rui Yan, Lili Mou, Yiping Song, Yansong Feng, and Dongyan Zhao. 2017. How to make context more useful? an empirical study on context-aware neural conversational models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–236, Vancouver, Canada. Association for Computational Linguistics.

G. Tur and R. D. Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Scott R Turner. 1994. Minstrel: A computer model of creativity and storytelling.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. 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 5998–6008.

Oriol Vinyals and Quoc V. Le. 2015. A neural conversational model. *CoRR*, abs/1506.05869.

Ellen M. Voorhees. 1999. The TREC-8 question answering track report. In *Proceedings of The Eighth Text REtrieval Conference, TREC 1999, Gaithersburg, Maryland, USA, November 17-19, 1999*.

Shengxian Wan, Yanyan Lan, Jiafeng Guo, Jun Xu, Liang Pang, and Xueqi Cheng. 2016. A deep architecture for semantic matching with multiple positional sentence representations. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2835–2841.

Yeyi Wang, Li Deng, and Alex Acero. 2011. Semantic frame-based spoken language understanding. *Spoken language understanding: systems for extracting semantic information from speech*, pages 41–91.

Jason D. Williams and Steve J. Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Comput. Speech Lang.*, 21(2):393–422.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive human-machine conversation with explicit conversation goal. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3794–3804, Florence, Italy. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 496–505, Vancouver, Canada. Association for Computational Linguistics.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3351–3357.

Chen Xing, Yu Wu, Wei Wu, Yalou Huang, and Ming Zhou. 2018. Hierarchical recurrent attention network for response generation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5610–5617.

Wei Xu and Alexander I. Rudnicky. 2000. Task-based dialog management using an agenda. In *ANLP-NAACL 2000 Workshop: Conversational Systems*.

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, Nanyun Peng, Ralph M. Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7378–7385.

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3740–3752, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018a. Commonsense knowledge aware conversation generation with graph attention. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4623–4629.

Kun Zhou, Wayne Xin Zhao, Yutao Zhu, Ji-Rong Wen, and Jingsong Yu. 2020. Improving multi-turn response selection models with complementary last-utterance selection by instance weighting. *CoRR*, abs/2002.07397.

Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. 2018b. Multi-turn response selection for chatbots with deep attention matching network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1118–1127, Melbourne, Australia. Association for Computational Linguistics.

Yutao Zhu, Zhicheng Dou, Jian-Yun Nie, and Ji-Rong Wen. 2020. Reboost: a retrieval-boosted sequence-to-sequence model for neural response generation. *Inf. Retr. J.*, 23(1):27–48.