Leveraging Narrative to Generate Movie Script

YUTAO ZHU, DIRO, UniversitÃĺ de MontrÃĺal, Canada
RUIHUA SONG*, Gaoling School of Artificial Intelligence, Renmin University of China, China
JIAN-YUN NIE, DIRO, UniversitÃĺ de MontrÃĺal, Canada
PAN DU, Thomson Reuters Labs, Canada, Canada
ZHICHENG DOU, Gaoling School of Artificial Intelligence, Renmin University of China, China
JIN ZHOU, Beijing Film Academy, China

Generating a text based on a predefined guideline is an interesting but challenging problem. A series of studies have been carried out in recent years. In dialogue systems, researchers have explored driving a dialogue based on a plan, while in story generation, a storyline has also been proved to be useful. In this paper, we address a new task–generating movie scripts based on a predefined narrative. As an early exploration, we study this problem in a "retrieval-based" setting. We propose a model (ScriptWriter-CPre) to select the best response (*i.e.*, next script line) among the candidates that fit the context (*i.e.*, previous script lines) as well as the given narrative. Our model can keep track of what in the narrative has been said and what is to be said. Besides, it can also predict which part of the narrative should be paid more attention to when selecting the next line of script. In our study, we find the narrative plays a different role than the context. Therefore, different mechanisms are designed for deal with them. 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. This new dataset is made available publicly to facilitate other studies in text generation under the guideline. 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.

CCS Concepts: • **Computing methodologies** → **Discourse, dialogue and pragmatics**.

Additional Key Words and Phrases: Narrative-guided Text Generation, Movie Script Generation, Retrieval-based Method

1 INTRODUCTION

A narrative is a series of related events or experiences, which can also be generally understood as a way of telling a story. The definition of narrative given by WordNet is "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] In natural language processing (NLP), narrative plays an important role in many tasks. For example, in story generation, the storyline can be treated as a type of narrative, which is helpful in generating coherent and consistent stories [8, 9].

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

*Ruihua Song is the corresponding author, and Gaoling School of Artificial Intelligence, Renmin University of China is the corresponding affiliation.
Authors' addresses: Yutao Zhu, yutao.zhu@umontreal.ca, DIRO, UniversitÃĺ de MontrÃĺal, MontrÃĺal, QuÃĺbec, Canada; Ruihua Song*, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China, rsong@ruc.edu.cn; Jian-Yun Nie, DIRO, UniversitÃĺ de MontrÃĺal, MontrÃĺal, QuÃĺbec, Canada, nie@iro.umontreal.ca; Pan Du, Thomson Reuters Labs, Canada, MontrÃĺal, QuÃĺbec, Canada, du@youark.com; Zhicheng Dou, Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China, dou@ruc.edu.cn; Jin Zhou, Beijing Film Academy, Beijing, China, whitezj@vip.sina.com.

1046-8188/2022/1-ART1 $15.00
https://doi.org/10.1145/3507356

Narrative	Jenny **doesn't like to go home**. To accompany Jenny, Gump decides to **go home later**. Gump is Jenny's **best friend**.	
Initial line	Mama's going to worry about me.	
1st line	Just **stay a little longer**.	✓
	Yeah, and I'll bet you $ 10,000 he laughs his ass off.	
2nd line	Ok, Jenny, **I'll stay**.	✓
	She lived in an old house.	
3rd line	He was a very loving man.	
	You are my most **special friend**.	✓

Fig. 1. An example of a part of a script with a narrative extracted from our *GraphMovie* dataset. The narrative generally describes the plot of a short session, which consists of several script lines. The checked lines are from a ground-truth session, while the unchecked ones are other candidates that are relevant but incoherent with the narrative. Our task is to select a proper line from a candidate list based on the given narrative and previous lines. By gradually selecting more lines, the whole session can be generated.

In dialogue generation, the narrative can act as a global plan for the whole conversation session, so as to avoid generating inconsistent and scattered responses [13, 46, 63]. In movie script generation, the narrative can act as a guideline to organize the plot in order to faithfully present the ideas elaborated in the narrative.

In this work, we investigate the utilization of narratives in the 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 as in general conversation, and hence can provide useful insight into a more general form of narrative-guided conversation or story generation. Our idea of using narrative as a guideline for script generation has two motivations: (1) **The use of narrative can make the generated script more consistent with previous script lines.** This has been demonstrated by existing studies that apply narratives or storylines as guidelines to conversation generation [8, 9]–Without the guidance of narratives, the utterances generated can easily deviate or become inconsistent, making the conversation useless and even unpleasant for the user. (2) **A narrative can provide a global view of what will happen in the plot so that the whole generated script lines can be more coherent.** The coherency of the generated texts is a problem not much addressed by the existing literature. Let us elaborate on the problem with a real example collected from the movie *Forrest Gump*. Given a narrative artificially written in several lines to retell a part of the plot in the movie, the corresponding conversation is shown in Figure 1. The first and third lines are spoken by Jenny, while the second line is given by Forrest. We can see that the narrative generally describes what happened in the several lines. If the narrative is not provided, the other choice of the second line ("She lived in an old house") can also be possible as it is consistent with the context (but it cannot lead to the following story). Therefore, it is important to leverage the narrative as a guideline for higher consistency and coherency of the generated movie script. This problem is very challenging, as has been recognized in some recent research [8, 9].

To alleviate the aforementioned problem, we study the problem of leveraging narrative to guide script generation, and we formulate our task as selecting the following lines by leveraging the narrative and previous lines. As an early exploration, we limit ourselves to the "retrieval-based" setting to simplify the task. To support our study on narrative-based generation, we collect a new dataset from GraphMovie, where end-users can retell the

story of a movie by uploading descriptive paragraphs in their own words.[2] More details about the dataset will be presented in Section 3.2.

The problem we address is closely related to dialogue generation that takes into account the context [60, 70, 72]. However, a narrative plays a different role from a general context. Particularly, a narrative covers a part of a story, and a good conversation guided by a narrative should cover all aspects it delivers, which is not the case for generating dialogue with a general context. Intuitively, there are two challenges in leveraging narratives: (1) One should keep track of the coverage of the information of the narrative to be aware of what has been said before. This role of narrative is different from that of context. In dialogue generation, typical methods of context-based response selection focus on measuring the matching degree between the context and the response. If the same matching mechanism is used on a narrative, one will often see redundant utterances. Narrative tracking aims to cope with this problem. (2) We need to determine which remaining part of the narrative should be expressed when generating the next immediate line. The narrative is often organized in a specific chronological order. Thus, the model for narrative-guided text generation needs to be able to learn how to use the remaining information in the narrative. In summary, it is necessary to design a new mechanism to track and leverage the narrative in the script generation/selection process.

In this paper, we propose a new model called **ScriptWriter-CPre** to address the problem of script selection with the help of a narrative. This model is extended from our proposed ScriptWriter, which can keep track of what in the narrative has been said and what is still remaining to select the next line by an updating mechanism. Matchings between the updated narrative, context, and response are then computed respectively and finally aggregated as a matching score. Although the narrative status has already been tracked by comparing the current context (history) with the narrative, it is still hard for the model to determine which remaining part to cover in the next line. This problem is challenging largely because of the lack of direct supervision signals. With only the final loss, it is hard to tell whether the wrong line stems from the mismatching with the context or the wrong usage of the narrative. To tackle this problem, we extend ScriptWriter with a **C**ontent **Pre**diction (CPre) module, which is inspired by recent studies on knowledge-driven dialogue [23, 28]. Specifically, we use the last line in the context to predict a distribution over the narrative, which is denoted as the "prior distribution". Then, we also predict a distribution based on the ground-truth line, which is denoted as the "posterior distribution". By reducing the distance between the two distributions, the model can learn to predict (select) the content from the narrative that should be covered by the next line. This is achieved by adding a supplementary KL-divergence loss to the final loss. The clear feedback from this loss function can directly help the model learn to use the narrative.

Existing work on composing conversations includes generation-based methods and retrieval-based methods. Due to the advantages of informative and fluent responses, retrieval-based approaches for conversation generation have been widely explored in previous studies [34, 60, 72, 74]. Additionally, retrieval-based methods provide an easier way for us to evaluate the impact of the narratives in our work. Therefore, we frame our work as a retrieval-based conversation generation task in terms of "generating" responses by selecting proper responses from a set of candidates.

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.

The problem we studied has several applications. Intuitively, our model can assist movie script authors to generate a new movie script. It can also be used for teaching, especially for beginners in movie creation. More generally, our method can be used in other text generation problems. For example, researchers have reported that generating a story from scratch is a very difficult problem. One of the solutions is providing the model with

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

a predefined storyline. Our method can be directly applied for this problem. Besides, similar approaches could also be applied to dialogue generation with a narrative or any type of guidance.

Our work has three main contributions:

(1) Our work is an early exploration of 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 has been made publicly accessible.

(3) We propose a new model in which a narrative plays a specific role in guiding script generation. Our model can not only track what in the narrative has already been covered, but also predict the part that should be expressed in the next line. This will be shown to be more appropriate than a general context-based approach.

(4) Extensive comparisons with nine baseline methods on both automatic and human evaluation metrics demonstrate the superiority of our proposed method consistently. The ablation studies further validate the effects of different modules in our design. The influences of different hyper-parameters, narrative types, and context lengths are also revealed in the experiments. Moreover, a case study and error analysis are performed, which provides us some insightful findings and inspires some promising future work.

The rest of the paper is organized as follows. Related work is introduced in Section 2. The problem formulation and the collection of the dataset are presented in Section 3. Then we describe the details of our method in Section 4, followed by the description of the experiments in Section 5. More analysis is conducted in Section 6. Finally, we conclude our paper and discuss future work in Section 7.

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 [36]. 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 [3, 49] or rule-based approaches to generate the narrative [37]. Recently, deep neural networks have been used to tackle the problem [1]. Related problems such as generating a coherent and cohesive narrative text [5, 7, 20], identifying relations in generated stories [39], and understanding relationship trajectories in narrative [68] 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 or how to generate a narrative). They did not investigate how to utilize the narrative in an application task such as dialogue generation.

2.2 Dialogue Systems

Human-computer conversation is one of the most challenging tasks in NLP. The target of this task is to produce replies for human input messages. Conversation systems have been built for both open-domain and domain specific dialogues. The open-domain dialogue systems aim to produce natural and human-like conversation without restriction of domains. A typical application is the chatbot. On the contrary, domain specific dialogue systems are often task-oriented, such as ticket booking systems and automatic diagnosis systems.

2.2.1 Open-domain chatbot. Inspired by the Turing test proposed in 1950 [48], researchers and engineers have developed many conversational systems for chitchat [6, 55]. ELIZA, created in 1966, is perhaps the first chatbot known publicly [55]. It can only chat with people in a specific domain based on many hand-crafted scripts and limited domain knowledge. Recently, with large-scale human conversational data on the Internet, researchers have explored data-driven approaches to building a chatbot. Existing methods can be categorized into two groups. The first group of approaches learn response generation from the data. Most of the early works are inspired by statistical machine translation [38]. In recent years, neural network-based models have been widely used. Based

on the sequence-to-sequence structure with attention mechanism [43, 51], multiple extensions have been made to tackle the "safe response" problem [26, 73]; to incorporate external knowledge [62, 71]; to generate responses with emotions or personas [27, 30, 35]; to model the hierarchical structure of the dialogue context [41, 42, 63] and to reduce the cost of response decoding [61]. Different from the generation-based methods, the second group of methods focus on searching for the most reasonable response from a large repository of conversational data. The key issue of these retrieval-based methods is how to measure the suitability of a response candidate for a user input. Early work studies single-turn response selection where the input is a single message [19, 21], while recent work pays more attention to context-response matching for multi-turn response selection. Representative methods include the deep learning to respond architecture (DL2R) [66], sequential matching network (SMN) [60], deep attention matching network (DAM) [72], deep utterance aggregation model (DUA) [70], interactive matching network (IMN) [15], and multi-hop selector network (MSN) [69]. 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.

2.2.2 Task-oriented systems. Different from open-domain chatbots, task-oriented systems are designed to accomplish tasks in a specific domain [25, 40, 47, 54]. In these systems, a dialogue state tracking component is designed for tracking what has happened in a dialogue [18, 57, 64]. This inspires us to track the information in the narrative that has not been expressed by previous lines of conversation. However, existing methods for task-oriented dialogue cannot be applied to our task directly as they are usually predefined for specific tasks, and state tracking is often framed as a classification problem. For example, in a movie-booking dialogue system, "Where do you want to watch a movie?" would fit into a predefined dialogue act slot *request(city)*, while the reply "I want to watch it in Seattle." is transformed into *inform(city="Seattle")* action [11]. Such state tracking and action design are impossible for our task because of the wide range of topics in movie scripts and corresponding narratives. To tackle this problem, we design a more general and flexible updating mechanism for the representation of the narrative, which keeps track of the information flow in movie dialogues.

2.3 Story Generation

Existing studies have also tried to generate a story. Early work relied on symbolic planning [4, 31] and case-based reasoning [12, 65], while more recent work uses deep learning methods. Some of them focused on story ending generation [16, 33], 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 [8, 9]. For example, compared with the former task, the latter is more challenging since the story framework and storyline should all be controlled by the model.

2.4 Other Forms of Text Generation

Some other text generation work is related to ours. Feng et al. [10] proposed an LSTM-based model to generate a paragraph-level text with multiple topics. The topic is represented by five words which contain far less information compared to the narrative we use. Their goal of writing a long text, *e.g.*, an essay, is also different from movie script generation. Kiddon et al. [22] proposed an interesting task of generating a recipe from given ingredients. The ingredients are represented by a checklist of phrases, and the model needs to incorporate them into the recipe in a reasonable order. They propose a checklist mechanism to update an agenda to control which ingredients have not been used yet. This idea is similar to our idea of updating the narrative, but the narrative used in our task consists of open-domain natural language sentences, whose vocabulary size is much larger than the structured list of limited ingredients.

Table 1. Statistics of *GraphMovie* corpus. A narrative is a description that summarizes a fragment of a movie. Each narrative corresponds to a session containing several lines of script. Micro-sessions are obtained by moving the prediction point through the session, each of which has a sequence of previous lines at that point of time, the same narrative as the session, and the next line to predict. The line candidates are used for prediction, which contain one golden line and several lines randomly sampled from the dataset.

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

Some recent studies also tried to guide the generation of dialogues [44, 59, 75] or stories [67] 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 the screenshots selected from the movie. Each movie on this website has, on average, 367 descriptions. A description 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 on IMDB as an initial list.[3] For each movie, we collect its descriptions 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 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 the corresponding description.

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

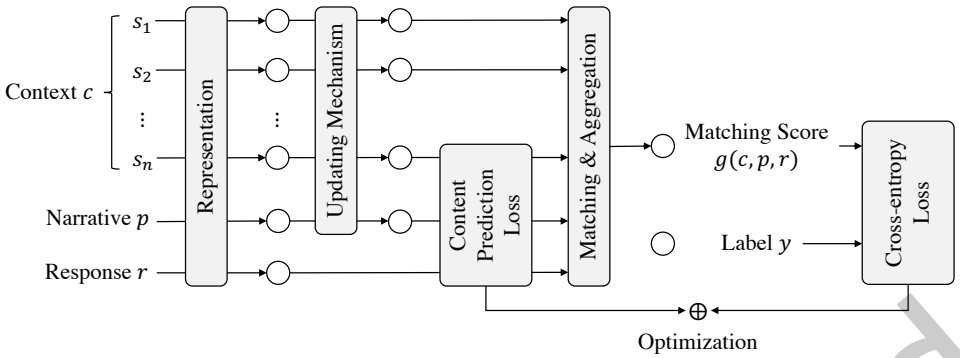


Fig. 2. The overview of our proposed ScriptWriter-CPre. ScriptWriter-CPre follows a representation-matching-aggregation framework. It is equipped with an updating mechanism to track what in a narrative has been expressed, based on which the representation of the narrative is updated. Besides, we also design a supplementary loss to facilitate the learning of content prediction.

As viewers of a movie can upload descriptions freely, not all descriptions 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 nonnarrative data. We also remove sessions that have less than two lines. Finally, we obtain 16,109 script sessions, each of which contains a description (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 the 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 [29, 60, 70] for retrieval-based dialogue.

4 PROPOSED METHOD: SCRIPTWRITER

4.1 Overview

A good response is required to be both coherent with the previous lines, *i.e.*, context, and 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 fragment 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-CPre to solve the problem. The overview of our model is shown in Figure 2. ScriptWriter-CPre 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, we propose using a supplementary loss to facilitate the learning of content prediction,

[4]Apache Solr, https://lucene.apache.org/solr/

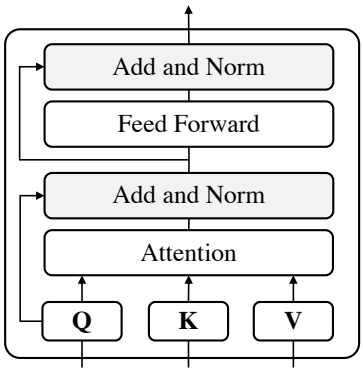


Fig. 3. The structure of the Attentive Block. The attention weights are computed by the query (**Q**) and key (**K**), and then applied to the value (**V**), **Q**, **K**, and **V** are three matrices. Residual connection, layer normalization and a feed-forward network are used to further fuse the information and output final results.

which lets the model learn to predict which part of the narrative should be given more attention in the next line. Fourth, 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). The whole model is optimized by the combination of the supplementary content prediction loss and the final response selection loss.

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 [50]. This structure is used to represent lines in the context, a narrative, and a response.

As illustrated in Figure 3, an attentive block contains two sub-layers: a sub-layer implementing an attention mechanism and a fully-connected feed-forward network (`FFN`). Each layer uses a residual connection to ease the training of networks [17] and layer normalization (`LayerNorm`) to avoid gradient vanishing and exploding [2]. The input of an attentive block consists of three sequence representations, namely query (**Q**), key (**K**), and value (**V**).[5] The output is a new representation of query and is denoted as $\texttt{AttentiveBlock}(\mathbf{Q}, \mathbf{K}, \mathbf{V})$ in the remaining parts. The detailed computation is as follows:

$$\texttt{AttentiveBlock}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \texttt{LayerNorm}(\mathbf{X} + \texttt{FFN}(\mathbf{X})), \tag{1}$$

$$\mathbf{X} = \texttt{LayerNorm}(\mathbf{Q} + \texttt{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})), \tag{2}$$

$$\texttt{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \texttt{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right) \cdot \mathbf{V}, \tag{3}$$

$$\texttt{FFN}(\mathbf{X}) = \texttt{FC}(\texttt{ReLU}(\texttt{FC}(\mathbf{X}))), \tag{4}$$

where $\texttt{FC}(\cdot)$ is the fully-connected layer, and $\texttt{ReLU}(\cdot)$ is the activation function. $\mathbf{Q} \in \mathbb{R}^{N_q \times d}$ denotes the attention query matrix, $\mathbf{K} \in \mathbb{R}^{N_k \times d}$ is the key matrix, and $\mathbf{V} \in \mathbb{R}^{N_k \times d}$ is the value matrix. N_q, N_k, and d are the number of

[5]Note that the "query" in the attention function and IR have different meanings.

attention query, key/value vectors, and the dimensions of the representation, respectively. The attentive block can be explained as follows: for each attention query vector in $\mathbf{Q}$, it first computes the dot product of the attention query with all keys, aiming to evaluate the similarity between the attention query and each key. Then, it is divided each by $\sqrt{d}$, and applies a softmax function to obtain the weights of the values. Finally, the new representation of the attention query vector is calculated as a weighed sum of values. The original Transformer framework is used for machine translation, where a self-attention is proposed to better represent a sentence. In our case, self-attention is used to recognize the internal connections between words in an utterance. So, query, key, and value become the same utterance. Our purpose of doing intra-sentence self-attention is to connect a word to related words in the sentence so that its representation can be enhanced by the related words in the higher layer.

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 looks up a pre-trained embedding table $\mathbf{E}$ and maps each word w into a d_e-dimension embedding $\mathbf{e}$:

$$\mathbf{e}_i^p = \texttt{LookUp}(w_i^p, \mathbf{E}), \quad \mathbf{e}_j^{s_i} = \texttt{LookUp}(w_j^{s_i}, \mathbf{E}), \quad \mathbf{e}_k^r = \texttt{LookUp}(w_k^r, \mathbf{E}). \tag{5}$$

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 $(l-1)^{\text{th}}$ 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:

$$\mathbf{P}^l = \texttt{AttentiveBlock}(\mathbf{P}^{l-1}, \mathbf{P}^{l-1}, \mathbf{P}^{l-1}), \tag{6}$$

$$\mathbf{S}_i^l = \texttt{AttentiveBlock}(\mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}), \tag{7}$$

$$\mathbf{R}^l = \texttt{AttentiveBlock}(\mathbf{R}^{l-1}, \mathbf{R}^{l-1}, \mathbf{R}^{l-1}), \tag{8}$$

where l ranges from 1 to L.

Inspired by previous studies [69, 72], we apply another group of attentive blocks, which is referred to as cross-attention, to capture the semantic dependency between p, s_i, and r. Considering p and s_i at first, their cross-attention representations are defined by:

$$\overline{\mathbf{P}}_{s_i}^l = \texttt{AttentiveBlock}(\mathbf{P}^{l-1}, \mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}), \tag{9}$$

$$\overline{\mathbf{S}}_{i,p}^l = \texttt{AttentiveBlock}(\mathbf{S}_i^{l-1}, \mathbf{P}^{l-1}, \mathbf{P}^{l-1}). \tag{10}$$

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 line and "go home later" in the narrative, become closer 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 $\overline{\mathbf{P}}_r^l$, $\overline{\mathbf{R}}_p^l$, $\overline{\mathbf{S}}_{i,r}^l$ and $\overline{\mathbf{R}}_{s_i}^l$:

$$\overline{\mathbf{P}}_r^l = \texttt{AttentiveBlock}(\mathbf{P}^{l-1}, \mathbf{R}^{l-1}, \mathbf{R}^{l-1}), \tag{11}$$

$$\overline{\mathbf{R}}_p^l = \texttt{AttentiveBlock}(\mathbf{R}^{l-1}, \mathbf{P}^{l-1}, \mathbf{P}^{l-1}), \tag{12}$$

$$\overline{\mathbf{S}}_{i,r}^l = \texttt{AttentiveBlock}(\mathbf{S}_i^{l-1}, \mathbf{R}^{l-1}, \mathbf{R}^{l-1}), \tag{13}$$

$$\overline{\mathbf{R}}_{s_i}^l = \texttt{AttentiveBlock}(\mathbf{R}^{l-1}, \mathbf{S}_i^{l-1}, \mathbf{S}_i^{l-1}). \tag{14}$$

These representations further provide matching information across different elements in the next step.

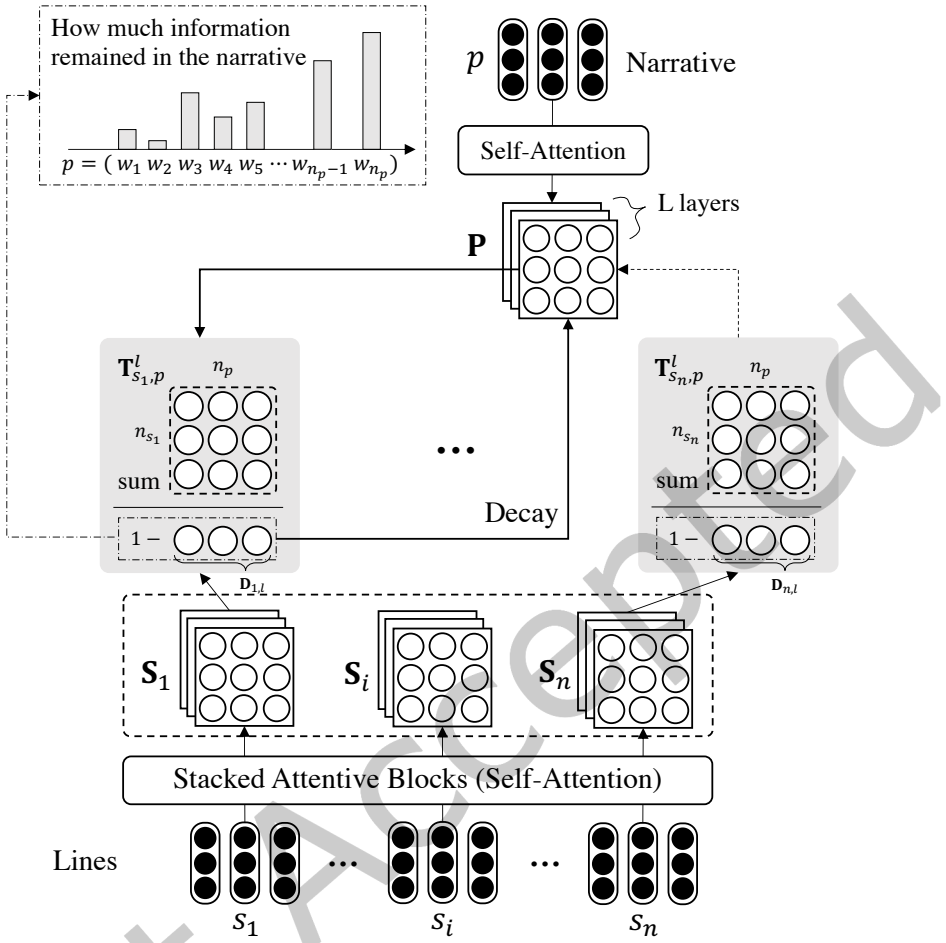


Fig. 4. Updating mechanism in ScriptWriter. The narrative and all lines in the context are first represented by the stacked attentive blocks. Then, the representation of the narrative is updated by lines in the context one by one. The information that has been expressed is decayed. As a result, the updated narrative focuses more on the remaining information.

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 4. We update a narrative's representation 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:

$$\mathbf{T}^l_{s_i,p}[j][k] = \cos(\mathbf{S}^l_i[j], \mathbf{P}^l[k]), \tag{15}$$

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

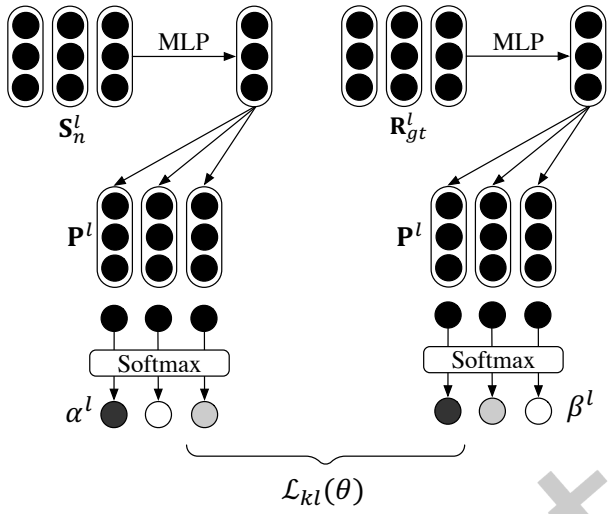


Fig. 5. A supplementary loss for the learning of content prediction. In the left side, for the the last line s_n, we compute its attention distribution over the narrative. This can be viewed as a "predicted" weight indicating which part of the content in the narrative should be focused on. To facilitate the prediction, in the right side, we use the ground-truth response r_{gt} to compute another attention distribution over the narrative. These attention weights reflect which part of the content in the narrative is indeed useful in selecting the ground-truth response. Therefore, we use the latter attention weights as the label and compute a KL-divergence loss to optimize the former predicted weights. By this means, the learning of content prediction in the narrative can be improved.

in the matching map $\mathbf{T}_{s_i,p}$. The summation at the l^{th} level is:

$$\mathbf{D}_i^l = [d_{i,1}^l, d_{i,2}^l, \cdots, d_{i,n_p}^l], \tag{16}$$

$$d_{i,k}^l = \gamma \sum_{j=1}^{n_{s_i}} \mathbf{T}_{s_i,p}^l[j][k], \tag{17}$$

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:

$$\mathbf{P}_{i+1}^l = (1 - \mathbf{D}_i^l)\mathbf{P}_i^l. \tag{18}$$

The initial representation $\mathbf{P}_0^l$ is equal to $\mathbf{P}^l$ defined in Equation (6). As a result, 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.

After the updating mechanism, the updated representation of the narrative will be used in the following operations.

4.4 Content Prediction

Inspired by recent studies on knowledge selection in knowledge-grounded dialogue [23, 28], we propose using a supplementary loss to facilitate the learning of content prediction (as shown in Figure 5). We define a task to predict the part of the narrative to which we should pay more attention in the next line. Given the last line s_n, we can predict a prior attention distribution over the narrative. However, this prior distribution cannot be

effectively learned as no "correct distribution" is given as supervision. To tackle this problem, we propose using a posterior distribution over the narrative to supervise the process. This posterior distribution is computed by the ground-truth response and the narrative, thus can provide valuable information for response selection.

Specifically, ScriptWriter-CPre first computes the sentence representation of the last line s_n:

$$\hat{\mathbf{S}}_n^l = \texttt{MLP}([\mathbf{S}_{n,1}^l, \cdots, \mathbf{S}_{n,n_{s_n}}^l]), \tag{19}$$

where $\mathbf{S}_{n,i}^l$ is the representation of the i^{th} word in s_n at the l^{th} layer of the attentive block. Then the prior attention distribution is computed as:

$$\alpha_i^l = \frac{\exp(\hat{\mathbf{S}}_n^l \cdot \mathbf{P}_i^l)}{\sum_{j=1}^{n_p} \exp(\hat{\mathbf{S}}_n^l \cdot \mathbf{P}_j^l)}, \tag{20}$$

where α_i^l is the attention weight of the i^{th} word in the narrative at the l^{th} layer. This distribution reflects how attention should be assigned to the narrative from the perspective of the last line.

Similarly, the posterior distribution can be computed with the ground-truth response r_{gt} as:

$$\beta_i^l = \frac{\exp(\hat{\mathbf{R}}_{gt}^l \cdot \mathbf{P}_i^l)}{\sum_{j=1}^{n_p} \exp(\hat{\mathbf{R}}_{gt}^l \cdot \mathbf{P}_j^l)}, \tag{21}$$

where $\hat{\mathbf{R}}_{gt}^l$ is the sentence representation of r_{gt} computed by the same process as Equation (19). This posterior attention distribution reflects which part of the narrative is useful for selecting the ground-truth response. So, it is desirable to distribute attention based on the posterior distribution. However, this latter is unknown during inference as no ground-truth response is given. Therefore, we propose to approximate the posterior distribution using the prior distribution so that ScriptWriter-CPre is capable of distributing appropriate attention even without posterior information. To this end, we introduce an auxiliary loss, namely the Kullback-Leibler divergence loss ($\mathcal{L}_{kl}$), to help optimize the model. The KL divergence loss is commonly used to measure the proximity between the prior distribution and the posterior distribution, which is computed as follows:

$$\mathcal{L}_{kl}(\theta) = -\sum_{(y,c,p,r)\in\mathcal{D}} \sum_{l=0}^{L} \sum_{i=1}^{n_p} \beta_i^l \log \frac{\beta_i^l}{\alpha_i^l}, \tag{22}$$

where θ denotes the model parameters.

When minimizing $\mathcal{L}_{kl}$, the posterior distribution β_i^l can be regarded as labels and ScriptWriter-CPre is trained to use the prior distribution α_i^l to approximate β_i^l. The representations of the last line and the narrative are directly used in computing the prior distribution. As a consequence, with $\mathcal{L}_{kl}$, the parameters of ScriptWriter-CPre, especially those for computing the last line's and narrative's representations, can be better tuned. It is worth noting that $\mathcal{L}_{kl}$ only works in training phase, thus the ground-truth response is not need in inference phase.

4.5 Matching

In the previous steps, we obtained the representations of all lines in the context, the representation of the response, and the updated representation of the narrative. In this step, we construct several matching maps and extract matching features based on these representations.

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 6.

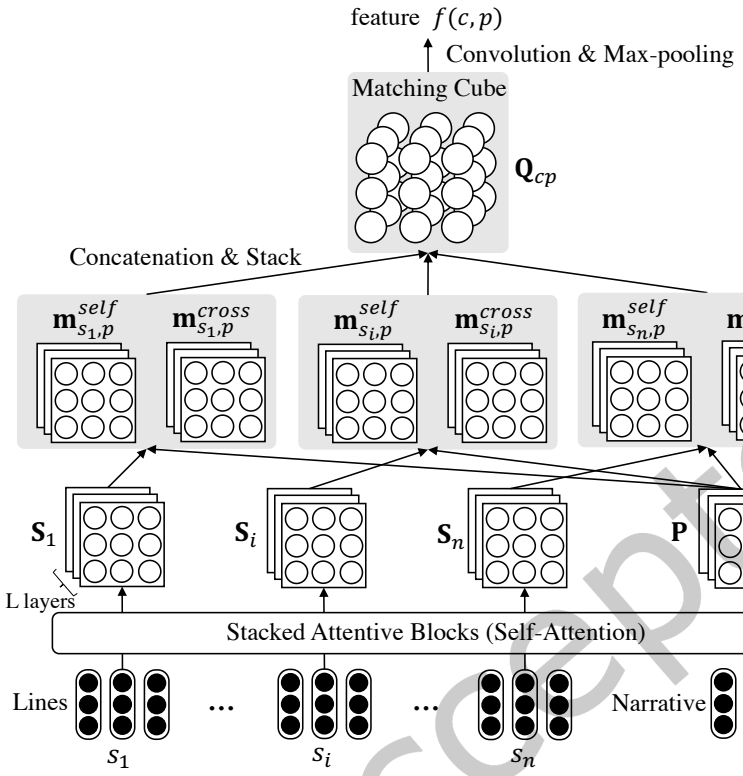


Fig. 6. The context-narrative matching. All lines and the narrative are represented by attentive blocks and the matching between them results in a matching cube $\mathbf{Q}_{cp}$. Matching features are extracted, aggregated, and distilled by a CNN with max-pooling operation.

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

$$\mathbf{m}^{self}_{s_i,p,l}[j,k] = \mathbf{S}^l_i[j]^\top \cdot \mathbf{P}^l[k], \tag{23}$$

$$\mathbf{m}^{cross}_{s_i,p,l}[j,k] = \overline{\mathbf{S}}^l_{i,p}[j]^\top \cdot \overline{\mathbf{P}}^l_{s_i}[k], \tag{24}$$

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

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

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

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

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

4.6 Aggregation

To further use the information across two consecutive lines, ScriptWriter piles up all context-narrative matching matrices and all 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 convolution 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 convolution 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:

$$f(c,p,r) = [f(c,p) \oplus f(c,r) \oplus f(p,r)], \tag{25}$$

$$g(c,p,r) = \texttt{sigmoid}\Big(\texttt{MLP}(f(c,p,r)\Big). \tag{26}$$

4.7 Model Training

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

$$\mathcal{L}_{ce}(\theta) = -\sum_{(y,c,p,r)\in\mathcal{D}} [y\log(g(c,p,r)) + (1-y)\log(1-g(c,p,r))]. \tag{27}$$

The two losses are combined to tune the model with a hyperparameter λ to control their effect:

$$\mathcal{L}(\theta) = \lambda\mathcal{L}_{ce}(\theta) + (1-\lambda)\mathcal{L}_{kl}(\theta). \tag{28}$$

5 EXPERIMENTS

5.1 Evaluation setup

As presented in Table 1, we randomly split the the *GraphMovie* collection into training, validation, and test sets. 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 [60, 70, 72]. We follow these previous studies and employ recall at position k in n candidates (R_n@k) and mean reciprocal rank (MRR) [52] as evaluation metrics. For example, R_{10}@1 means recall at one when we rank ten candidates (one positive sample and nine negative samples). The final results are the average numbers over all micro-sessions in the test set. Note that among different Recall metrics, R_2@1

and $R_{10}@1$ are two most relevant ones to our task, because there is only one positive line in our dataset. They correspond to the scenarios in the training and test sets, respectively.

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 selections divided by the number of lines in a session. We consider two measures: 1) P_{strict} which accepts a right response at the right position; 2) P_{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 a fully-connected layer to produce a final score. This second approach turns out to perform better. Therefore, we only report the results with this latter method.[6]

(1) MVLSTM [53]: this model concatenates all previous lines as a context document and leverages an LSTM to obtain positional representations for all words in the document and the response candidate. Then the interactions between them at different positions are modeled by cosine similarity, resulting in a matching map. The matching features are extracted with a k-max pooling layer and aggregated as a matching score with an MLP. To incorporate a narrative into this model, we conduct the same operation to compute the matching score between the narrative and a response candidate. Finally, the two scores are combined as a final matching score with another MLP. This model considers all positional information in the sentence, but only the top k values in the matching map are used.

(2) DL2R [66]: the model reformulates the last line with other lines in a context with multiple approaches. The reformulated line and a response candidate are then represented by a composition of an RNN and a CNN. The matching score is computed in a similar way as `MVLSTM`. We use the same RNN and CNN to represent the narrative and compute a matching score between the context and the narrative, which is further combined with the context-response matching score to output a final score with an MLP. In this model, all previous lines in the context are used to reformulate the last line, thus the context-response matching is neglected.

(3) SMN [60]: it matches each line with the response sequentially, and then transforms each line-response pair into a matching vector with CNNs. The matching vectors are aggregated with an RNN as a matching score of the context and the response candidate. We apply the same CNNs to obtain a matching vector for each line-narrative pair and use an RNN to compute a matching score. Two matching scores are finally combined by an MLP. This model is not equipped with an attention mechanism, which may provide better representations.

(4) DAM [72]: it represents a context and a response by conducting a self-attention and a cross-attention operation on them. It uses CNNs to extract features and uses an MLP to get a score. Similar to SMN, we perform the same operation to obtain a matching score of the context and the narrative and combine the score with context-response matching score by an MLP. 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.

[6]We also tested some basic models such as RNN, LSTM, and BiLSTM [29] in our experiments. However, they cannot achieve comparable results to the selected baselines.

(5) DUA [70]: the model concatenates the last line with each previous line in the context and response, respectively. Then it performs a self-attention operation to get the refined representations for both the context and the response, based on which matching features are extracted with CNNs and RNNs. Finally, it uses an MLP to get a matching score. We apply the same self-attention operation to get the refined representation of a narrative. Then we extract the matching features between the refined response and the narrative. Finally, both groups of matching features are aggregated by an RNN to get a final score. This model uses RNNs to represent sentences and aggregate the matching information, which is different from our model.

(6) IMN [15]: the model uses an attentive hierarchical recurrent encoder, which is capable of encoding sentences hierarchically and aggregating them with an attention mechanism to produce more descriptive representations. Then the bidirectional interactions between the whole multi-turn context and the response candidates are calculated to derive the matching information between them. We apply the same structure to derive the matching features between the narrative and the response. These features are fused with context-response matching features to output a final score with an MLP.

(7) IOI [45]: the model performs matching by stacking multiple interaction blocks in which the residual information from one step of interaction initiates the interaction process again. Therefore, the matching information within a line-response pair is extracted from the interaction of the pair iteratively, and the information flows along the chain of blocks via representation. To leverage the narrative, we use the same multiple interaction blocks to extract matching information within the narrative-response pair and compute a matching score. The two scores are added together to select the response.

(8) MSN [69]: this model first utilizes a multi-hop selector to select the relevant lines as context. Then, the model matches the filtered context with the candidate response based on their self-attention and cross-attention representations. Next, the matching features are extracted by a CNN and aggregated by an LSTM. The final matching score is computed by an MLP. The matching process is similar to SMN, thus we use a similar way to incorporate the narrative.

(9) ScriptWriter [76]: This is the previous model we proposed. This model does not have the narrative representation optimization module, and the other structure is similar to ScriptWriter-CPre proposed in this paper.

5.3 Training Details

All models are implemented in Tensorflow.[7] Word embeddings are pre-trained by Word2vec [32] 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 both have 32 filters, respectively. They both use [3,3,3] as the kernel size, and the max-pooling size is [3,3,3]. Two 2D convolutional layers of narrative-response matching both have 32 filters with [3,3] as the kernel size. The max-pooling size is also [3,3]. All parameters are optimized with Adam optimizer [24]. The learning rate is 0.001 and decreases during training. The initial value of 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 the 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.[8]

[7] Tensorflow, https://www.tensorflow.org
[8] Our project, https://github.com/DaoD/ScriptWriter

Table 2. Evaluation results on two response selection tasks: turn-level and session-level. The turn-level evaluation aims at measuring the performance of models on predicting a specific line in a session, while the session-level evaluation considers the quality of the whole session. † and ⋆ denote significant differences between each baseline and ScriptWriter-CPre measured in t-test with $p \leq 0.01$ and $p \leq 0.05$, respectively.

	Turn-level					Session-level	
	$\mathbf{R_2@1}$	$\mathbf{R_{10}@1}$	$\mathbf{R_{10}@2}$	$\mathbf{R_{10}@5}$	**MRR**	$\mathbf{P_{strict}}$	$\mathbf{P_{weak}}$
MVLSTM	$0.651^{\dagger}$	$0.217^{\dagger}$	$0.384^{\dagger}$	$0.732^{\dagger}$	$0.395^{\dagger}$	$0.198^{\dagger}$	$0.224^{\dagger}$
DL2R	$0.643^{\dagger}$	$0.210^{\dagger}$	$0.321^{\dagger}$	$0.638^{\dagger}$	$0.314^{\dagger}$	$0.230^{\dagger}$	$0.243^{\dagger}$
SMN	$0.641^{\dagger}$	$0.176^{\dagger}$	$0.333^{\dagger}$	$0.696^{\dagger}$	$0.392^{\dagger}$	$0.197^{\dagger}$	$0.236^{\dagger}$
DAM	$0.631^{\dagger}$	$0.240^{\dagger}$	$0.398^{\dagger}$	$0.733^{\dagger}$	$0.408^{\dagger}$	$0.226^{\dagger}$	$0.236^{\dagger}$
DUA	$0.654^{\dagger}$	$0.237^{\dagger}$	$0.403^{\dagger}$	$0.736^{\dagger}$	$0.396^{\dagger}$	$0.223^{\dagger}$	$0.251^{\dagger}$
IMN	$0.686^{\dagger}$	$0.301^{\dagger}$	$0.450^{\dagger}$	$0.759^{\dagger}$	$0.463^{\dagger}$	$0.304^{\dagger}$	$0.325^{\dagger}$
IOI	$0.710^{\dagger}$	$0.341^{\dagger}$	$0.491^{\dagger}$	$0.774^{\dagger}$	$0.464^{\dagger}$	$0.324^{\dagger}$	$0.337^{\dagger}$
MSN	$0.724^{\dagger}$	$0.329^{\dagger}$	$0.511^{\dagger}$	$0.794^{\star}$	$0.464^{\dagger}$	$0.314^{\dagger}$	$0.346^{\dagger}$
ScriptWriter	$0.730^{\dagger}$	$0.365^{\dagger}$	0.537	0.814	0.503	0.373	$0.383^{\star}$
ScriptWriter-CPre	**0.756**	**0.398**	**0.557**	**0.817**	**0.504**	**0.392**	**0.409**

5.4 Evaluation Results

5.4.1 Automatic Metrics. 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 MSN and IOI, which are two state-of-the-art models for multi-turn response selection. Most improvements are statistically significant (p-value ≤ 0.01). MSN and DAM perform better than other baselines, which confirms the effectiveness of the self- and cross-attention mechanism used in this model. IOI, IMN, and DUA also apply the attention mechanism. They outperform the other baselines that do not use attention. Both observations confirm the advantage of using attention mechanisms over pure RNN (such as SMN, DL2R, and MVLSTM).

In terms of session-level evaluations, ScriptWriter-CPre achieves 1.9% and 2.6% absolute improvements over the best results obtained by the baseline methods. This demonstrates that ScriptWriter-CPre can generate a more coherent and consistent script. Besides, between the two session-level measures, we observe that both our ScriptWriter-CPre and ScriptWriter are less affected when moving from P_{weak} to P_{strict}. This shows that the two models can better select a response at the right position. We attribute this behavior to the utilization of narrative coverage.

As a retrieval-based method, the selected response (*i.e.*, the one with the highest score) is used for constructing the session, so R@1 is the most important metric. Comparing ScriptWriter-CPre with ScriptWriter, we can see the content prediction module can improve both $R_2@1$ and $R_{10}@1$ significantly. Correspondingly, we can see the performance on session-level evaluation is also improved. These results demonstrate the effectiveness of our proposed content prediction module. We also notice that MRR is only improved by 0.001. This means that, for those samples where the ground-truth response cannot be ranked at the top, the average position of the ground-truth response decreases. From this result, we can speculate that our proposed supplementary loss in content prediction module is helpful when the future lines can be inferred from the last line and the narrative. In the future, we plan to consider more script lines in the context for content prediction.

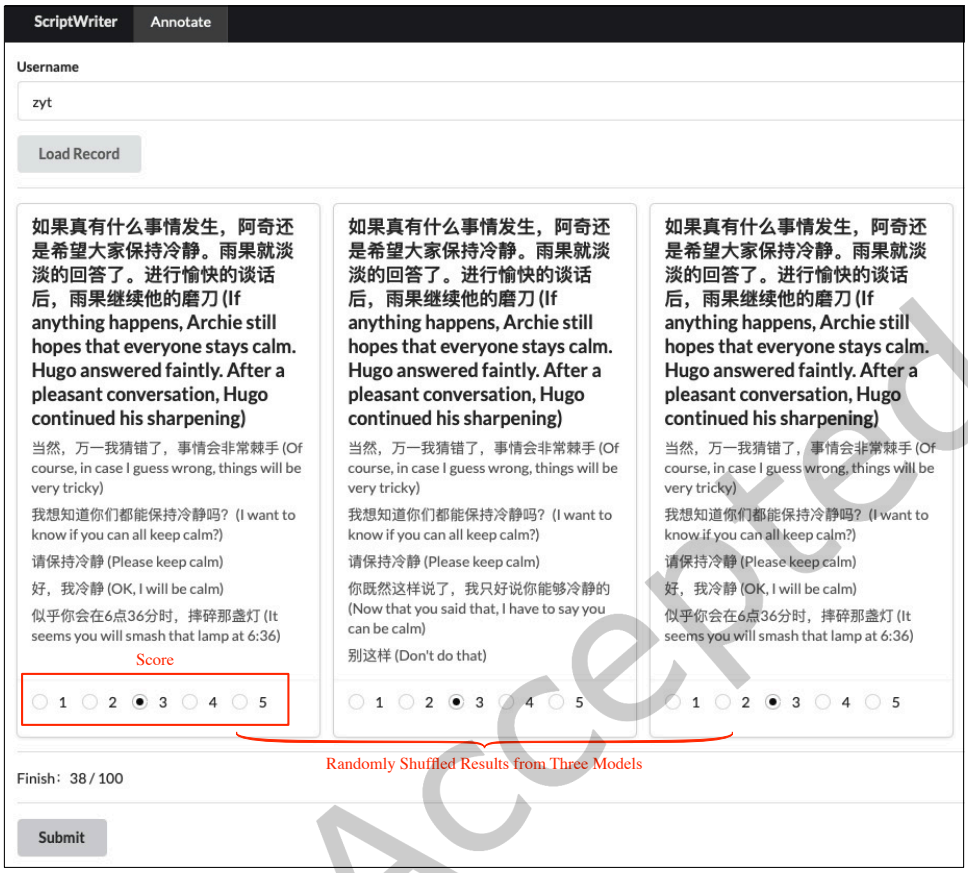


Fig. 7. Illustration of the annotation system. Three sessions generated by different models (ScriptWriter-CPre, ScriptWriter, and MSN) are randomly shuffled and presented to the annotators. The annotators should label them with a score ranging from 5 (the best) to 1 (the worst). In total, 100 generated sessions with different narratives are annotated.

Besides, it is worth noting that the content prediction module only adds one MLP layer (with a few parameters) to the model, which has less influence on the training cost. However, this has improved the model's performance, especially on session-level evaluation. This suggests that this module is beneficial for our model.

5.4.2 Human Evaluation. Similar to dialogue generation, the ground-truth script lines are not the only suitable ones for a given narrative due to the diversity of human language. Therefore, we also conduct a human evaluation to compare our method with previous baselines. We hire four experienced annotators to annotate 100 generated sessions of MSN, ScirptWriter, and ScriptWriter-CPre. The narrative is randomly selected from the test set, and the first line is also given. As shown in Figure 7, the four annotators are asked to label the randomly shuffled results with scores ranging in [1, 5]. The standard of scoring is based on the errors appearing in session coherence and narrative consistence, which is shown as follows:

- 5: The generated lines are coherent and consistent with the narrative perfectly (0 error in coherence, 0 error in consistence).

Table 3. Examples of annotation results. For each narrative, the results from three different models (ScriptWriter-CPre, ScriptWriter, and MSN) are annotated. The labels given by a human annotator are shown in the last column of the table, which reflect the quality of the session.

Narrative: Maxie stopped the noodles at the entrance of the hospital and said that he would also go with him. Every time they have conflicts, they talk about swimming.		
ScriptWriter-CPre	(1) Hey! (2) Hey, let me go first. (3) I thinkâĂę I should also go the beach. (4) Do you want to go swimming? (5) Yeah, letâĂŹs go.	5
ScriptWriter	(1) Hey! (2) Hey, let me go first. (3) I thinkâĂę I should also go the beach. (4) Do you want to go swimming? (5) It doesn't matter if you don't want to go. You can wait another five or six years.	4
MSN	(1) Hey! (2) Hey, let me go first. (3) Hey, IâĂŹm Lucas. (4) Do you want to go swimming? (5) Yeah, letâĂŹs go. (6) I appreciate God every day.	2
Narrative: Bernstein asked the reporter to talk to Li Lan. Li Lan and Kane had different views on the Spanish-American War.		
ScriptWriter-CPre	(1) You should meet Jack Lilan. (2) Of course he and Mr. Kane. (3) Miss Tang Sheng, this is Mr. Kane. (4) Ask you to kill somebody.	3
ScriptWriter	(1) You should meet Jack Lilan. (2) Of course he and Mr. Kane. (3) Mr. Kane, I have something to look for you. (4) A little bit of disagreement.	3
MSN	(1) You should meet Jack Lilan. (2) Of course he and Mr. Kane. (3) Miss Tang Sheng, this is Mr. Kane. (4) Ask you to kill somebody. (5) I have one thing to say.	2
Narrative: Hickler invited her to dance together, and Napaloni's wife was very happy.		
ScriptWriter-CPre	(1) Nobody? IâĂŹm honored to invite you. (2) IâĂŹm honored too, please sit down and welcome. (3) Are you inviting me?	4
ScriptWriter	(1) Nobody? IâĂŹm honored to invite you. (2) IâĂŹm honored too, please sit down and welcome. (3) Are you inviting me? (4) I love music very much especially Beethoven. (5) Thank you very much. (5) I'm always happy to meet you.	2
MSN	(1) Nobody? IâĂŹm honored to invite you. (2) Fahan Laiju will come to my office tomorrow to see me. (3) Are you inviting me? (4) Ok, let me do it myself.	4

- 4: The generated lines are coherent but have one line that is not consistent with the narrative (0 error in coherence, 1 error in consistence).
- 3: There is one line that is not coherent with the context and at most one line is not consistent with the narrative (1 error in coherence, $\leq$ 1 error in consistence).
- 2: There is at most one line that is incoherent with the context and more than one line are inconsistent with the narrative ($\leq$ 1 error in coherence, $\geq$ 1 error in consistence).
- 1: There is more than one line that are incoherent or inconsistent with the narrative ($\geq$ 1 error in coherence, $\geq$ 1 error in consistence).

Some annotation results are shown in Table 3.

Following the recent work [58], we compute the Kendall tau-b correlation coefficient to evaluate the agreement between any two annotators, and then we average the Kendall tau scores over the 100 samples and six pairs of

Table 4. Human evaluation results. The quality of the whole generated session is evaluated.

(a) Absolute scoring. Annotators are asked to score the generated sessions from different models, respectively. The scores range from 5 (the best) to 1 (the worst). The average score reflects the overall performance of each model.

Model	1	2	3	4	5	Average
MSN	**13.50%**	**17.50%**	19.50%	23.00%	26.50%	3.3150
ScriptWriter	10.75%	15.75%	23.00%	**26.00%**	24.50%	3.3775
ScriptWriter-CPre	8.50%	13.50%	**25.50%**	24.25%	**28.25%**	**3.5025**

(b) Relative preference scoring. Annotators are asked to compare the results from two different models, and label which one is better. "Tie" indicates the quality of the generated sessions from two models are similar.

	Win	Tie	Lose
ScriptWriter-CPre vs. ScriptWriter	27.75	51.5	20.75
ScriptWriter-CPre vs. MSN	33.25	47.0	19.75
ScriptWriter vs. MSN	28.25	47.5	24.25

annotation results. The coefficient is 0.633, which indicates that annotators have moderate agreement on the scoring order of the generated script lines.

The evaluation results are shown in Table 4(a), which are consistent with the automatic evaluation results in general. It is clear that ScriptWriter-CPre performs the best (3.5 score on average) among the three models. More specifically, 28.25% of the results generated by our ScriptWriter-CPre are perfect and 78% are scored higher than 2. These results demonstrate the effectiveness of our proposed method. MSN has 31% results with 1 or 2 score, which is the worst among the three models. Besides, we also run a pair-wise comparison between each pair of models, and this results in Table 4(b). The relative preference scores also show that both ScriptWriter-CPre and ScriptWriter are preferred to MSN. This indicates that using a narrative to guide the generation of script is a much more appropriate approach than using it as a part of the general context. Between ScriptWriter-CPre and ScriptWriter, ScriptWriter-CPre wins, indicating that the extension we propose in this paper can further improve the approach.

6 FURTHER ANALYSIS

6.1 Model Ablation

We conduct an ablation study to investigate the impact of different modules in ScriptWriter-CPre. These studies are conducted from different perspectives:

- We investigate the influence of the number of the layers by setting $l = \{1, 2, 3\}$;
- We validate the effectiveness of our proposed updating mechanism. Specifically, we set $\gamma = 0$ and test the performance of our model. Under this circumstance, the representation of the narrative is not updated but static;
- We test the influence of the cross-attention representations used in our method;
- We explore the effectiveness of narrative-response, context-narrative, and context-response matching by removing them one by one from the entire model;
- Finally, we investigate the influence of our proposed supplementary loss $\mathcal{L}_{kl}$ for narrative representation optimization. This is achieved by setting $\lambda = 1$.

Table 5. Ablation results in two response selection tasks: turn-level and session-level. We test our model with different layers of attention blocks. Besides, we also validate the effectiveness of different modules by removing them one by one from our model. These include the updating mechanism for the narrative representation, the cross-attention mechanism, the narrative-response (PR), context-narrative (CP), and the context-response (CR) matching, and the supplementary loss for learning content prediction.

	Turn-level					Session-level	
	$\mathbf{R_2@1}$	$\mathbf{R_{10}@1}$	$\mathbf{R_{10}@2}$	$\mathbf{R_{10}@5}$	**MRR**	$\mathbf{P_{strict}}$	$\mathbf{P_{weak}}$
ScriptWriter-CPre	**0.756**	**0.398**	**0.557**	**0.817**	**0.504**	**0.392**	**0.409**
0 layer	0.727	0.356	0.525	0.791	0.475	0.346	0.364
1 layer	0.729	0.359	0.533	0.806	0.490	0.358	0.368
2 layers	0.735	0.374	0.541	0.814	0.493	0.368	0.384
w static narrative	0.736	0.367	0.537	0.814	0.495	0.357	0.371
w/o Cross	0.720	0.365	0.532	0.809	0.500	0.357	0.361
w/o PR matching	0.650	0.242	0.401	0.720	0.387	0.210	0.230
w/o CP matching	0.734	0.381	0.537	0.809	0.495	0.375	0.391
w/o CR matching	0.727	0.340	0.493	0.768	0.469	0.361	**0.419**
w/o $\mathcal{L}_{kl}$	0.729	0.367	0.538	0.816	0.496	0.370	0.378

Model ablation results are shown in Table 5. We have the following findings:

(1) In general, the more layers used, the higher the performance of ScriptWriter-CPre. This result is consistent with existing work, which also shows that multi-layer structures are preferred [45, 72]. It is possible that more than three layers can yield even better results. However, due to the computational resources we have, we can only test at most three layers in the experiments. Consumed memory and time increase along with the number of layers. This suggests that adding more layers is a good strategy only if we have the necessary computation power.

(2) ScriptWriter-CPre performs better than it with static narrative representation, demonstrating the effectiveness of the updating mechanism for the narrative. The optimal value of γ is around 0.647 after training, which means that only about 35% of the information in the narrative is kept when a line conveys it.

(3) When cross-attention representations are removed from ScriptWriter-CPre, the performance greatly degrades. This indicates that the cross-attention mechanism can capture useful interaction features between different sources of information such as narrative and response.

(4) 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.

(5) 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.

(6) 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.

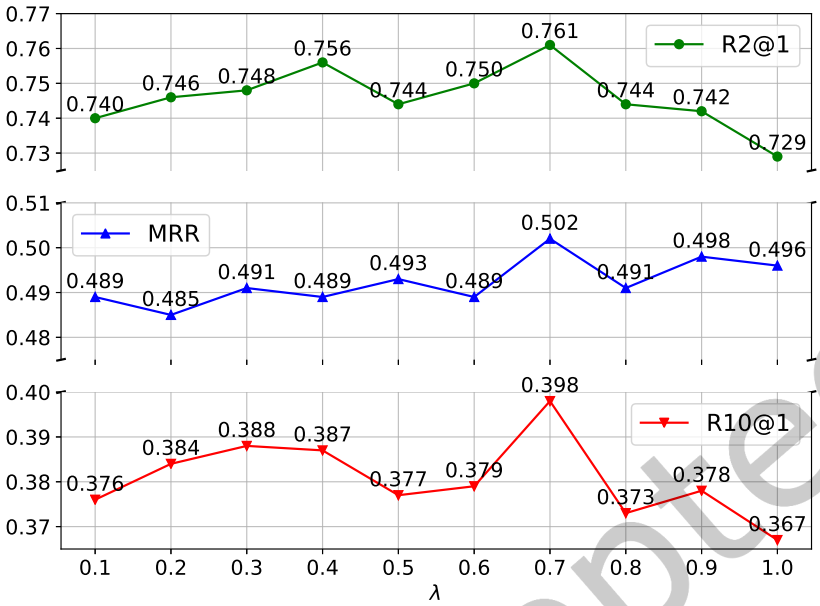


Fig. 8. The performance of ScriptWriter-CPre on the validation set with different λs.

(7) When removing the supplementary loss $\mathcal{L}_{kl}$, the model is similar to the original ScriptWriter proposed in [76].[9] We can see the performance drops significantly in both turn-level and session-level evaluations. This result demonstrates the effectiveness of $\mathcal{L}_{kl}$ in optimizing the representation of the narrative.

6.2 Performance with Different λs

To investigate the impact of supplementary loss $\mathcal{L}_{kl}$ in our model, we vary λ from 0.1 to 1.0 and report R_2@1, R_{10}@1, and MRR results on the validation set in Figure 8. Generally, the performance increases along with the increase of λ and achieves the best at around $\lambda = 0.7$. Thereafter, the performance starts to decrease. When λ is very small, the model is optimized mainly according to $\mathcal{L}_{kl}$. The performance is not good because the main task, generating script lines, cannot be well-learned. With increased λ, the main task plays a more important role in the optimization, so that the performance improves. However, when $\lambda > 0.7$, the performance start to decrease. This is because $\mathcal{L}_{kl}$ can no longer play a significant role with a very small weight. When $\lambda = 1$, no supplementary loss is used, and we have analyzed this result (at the last point in Section 6.1).

6.3 Performance across Narrative Types

As we explained, the narratives in our dataset are contributed by Internet users, 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 9.

[9]There are some minor differences in hyperparameters, such as a different number of convolutional filters, leading to slightly different results than ScriptWriter reported in Table 5.

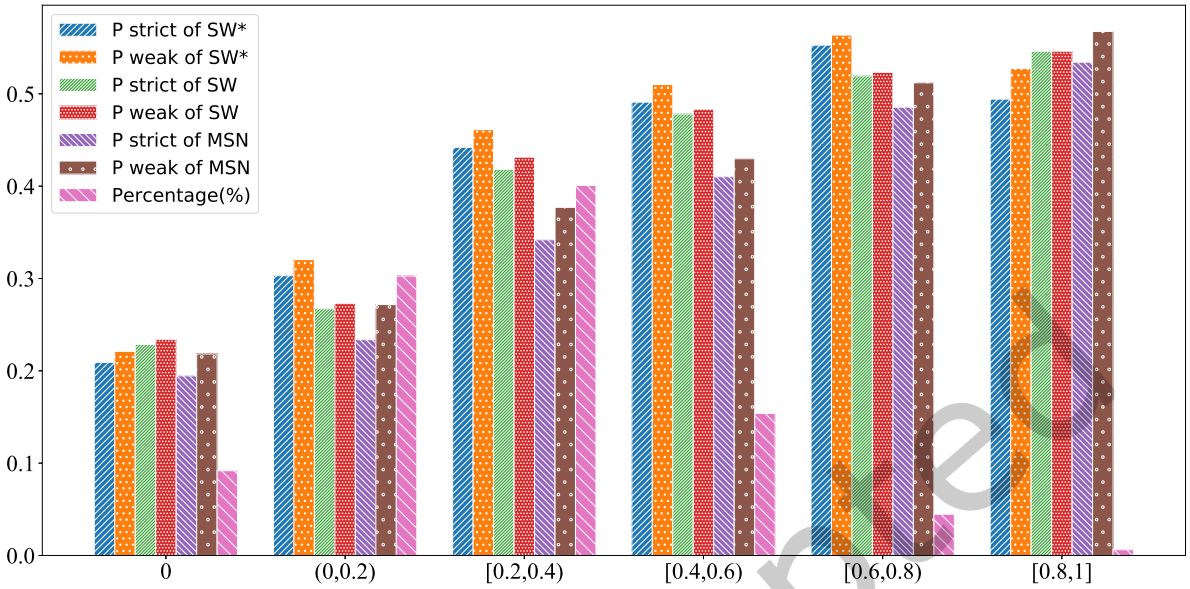


Fig. 9. The performance of ScriptWriter-CPre (SW*), ScriptWriter (SW), and MSN on the test set with different types of narrative in session-level evaluation.

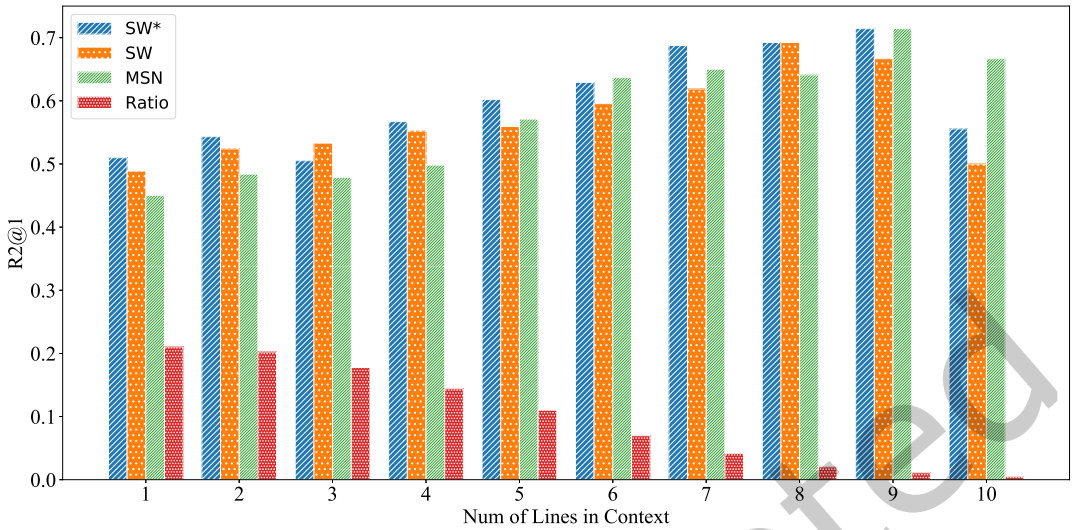
We plot the performance of ScriptWriter-CPre, ScriptWriter, and MSN in session-level evaluation over different types of narratives. We have the following observations:

(1) 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 all models perform poorly on this type compared to other types in terms of P_{strict}. Moreover, we find that ScirptWriter outperforms ScirptWriter-CPre on this type of data. The difference between these two models is that ScriptWriter does not apply the supplementary loss for narrative representation learning. The results reflect that our proposed supplementary loss cannot perform well when there are fewer overlapping words between the narrative and the lines. The reason is that, under this circumstance, it is hard to compute the similarity between the narrative and the line. Therefore, the loss $\mathcal{L}_{kl}$ is less accurate.

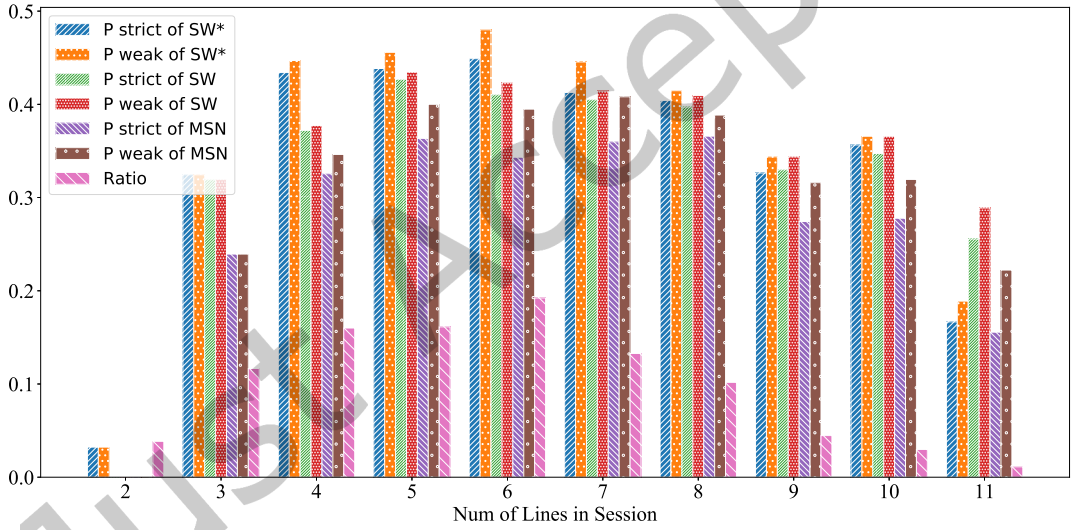
(2) The performance tends to become better when the overlap ratio is increased. This is consistent with our intuition: when a narrative is more detailed and better aligned with the session in wording, it is easier to choose the best responses. This plot also shows that both ScriptWriter-CPre and ScriptWriter can achieve better performance than MSN on all types of narratives, which further demonstrates the effectiveness of using narrative to guide the dialogue.

(3) Interestingly, we find that ScriptWriter-CPre performs worse than the other two models in buckets "[0.8, 1]". The potential reason is that there are a lot of words shared between the narrative and the lines, so the model can capture their relationships easily, and the supplementary loss has less effect. However, we cannot draw a solid conclusion on this as there are only 0.6% data lying in this bucket.

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



(a) Turn-level evaluation with $R_2@1$.



(b) Session-level evaluation.

Fig. 10. The performance of ScriptWriter-CPre (SW*), ScriptWriter (SW), and MSN on the test set with different number of lines. We show the performance in both turn-level and session-level evaluation.

6.4 Performance with Various Context Lengths

We study how ScriptWriter-CPre performs in contexts of different lengths and compare it with ScriptWriter and MSN at both turn level and session level.

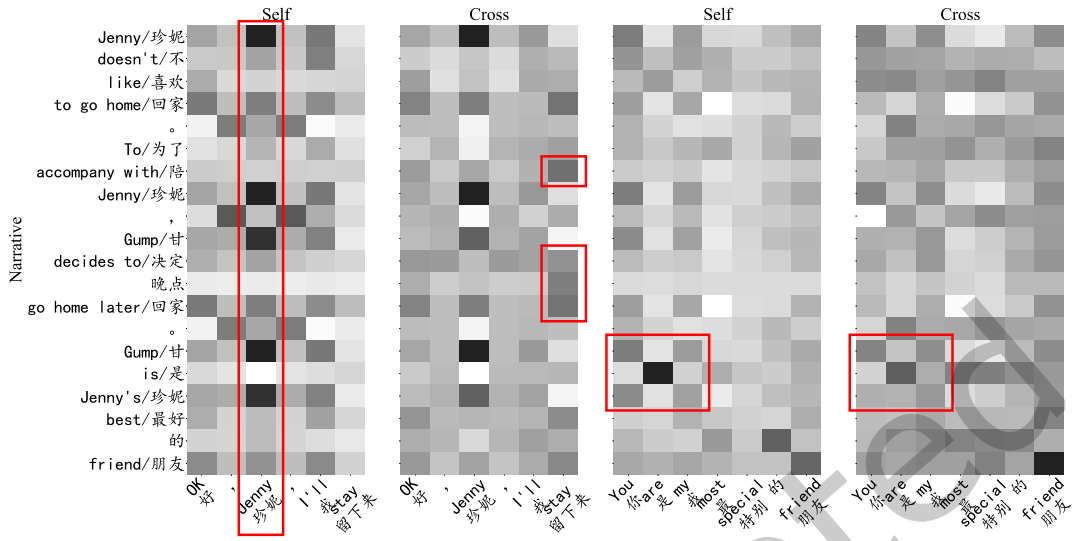


Fig. 11. Matching maps between the response (the second turn in the case study) and the given plan in the first and second level, namely $\mathbf{m}_{p,r,0}^{self}$ and $\mathbf{m}_{p,r,1}^{self}$.

The results of the turn-level evaluation are shown in Figure 10a. At first, it is clear that the performance of all models increases along with the increase of the number of lines of context from one to eight. This is because more lines in the context can provide more information, which help the model to select better responses. Also, ScriptWriter-CPre performs much better than MSN on very short context. This demonstrates that our method can handle the case even when the matching information is insufficient. Finally, when there are more than nine lines in the context, the performance of all models decreases. This could be due to the fact that too many lines of context may more likely contain irrelevant information for the current turn. However, this part of the data only contain less than 2% samples. We will study the problem in the future when more data are available.

The evaluation results at the session level are illustrated in Figure 10b. The x-axis is the total number of lines in the session. Different from the results at turn level, all models perform best with around six to eight lines. The reasons include: (1) When there are only a few lines in the context, the context-response matching, which is essential in our model as shown in the ablation study, cannot perform well, thus limiting the model to selecting a proper response. (2) When a session contains many lines of script, intuitively, the narrative is also long. A long narrative usually contains a lot of points, which should be covered accordingly by the lines. Therefore, these cases are much harder for the model to capture. Nevertheless, our model outperforms MSN in almost all cases, which further demonstrates its effectiveness.

6.5 Visualization

We visualize four attention maps between a narrative and two consecutive responses in both self-attentive blocks and cross-attentive blocks ($\mathbf{m}_{p,r,0}^{self}$ and $\mathbf{m}_{p,r,0}^{cross}$) to see what information the attentive mechanism tries to capture.

From Figure 11, we show the heat map of attention for the example given in the introduction section. It is interesting to see that only some shared words, like “Jenny” and “Jenny”, or highly co-occurred words, like “Jenny” and “Gump”, can have high similarity in the self-attention matching, while the semantically related words, like “stay” and “go home later” or “accompany with”, can have higher similarity in the cross-attention matching.

Narrative	Jenny doesn't like home①. To accompany with Jenny, Gump decides to go home later②. Gump is Jenny's best friend.③			
Initial line	Mama's going to worry about me.			
Model	Ground-truth	ScriptWriter-CPre (Our model)	ScriptWriter	MSN
1st line	Just stay a little longer.	Just stay a little longer. ①	Just stay a little longer.①	Go file it.
2nd line	Ok, Jenny, I'll stay.	OK, Jenny, I'll stay.②	OK, Jenny, I'll stay.②	OK, Jenny, I'll stay.②
3rd line	You are my most special friend.	Forrest, I do love you.③	Forrest!	You are my most special friend.③
4th line	EOS	The best friend.③	The best friend.③	The best friend.③

Fig. 12. A case study. These are some results sampled from the session-level response selection task. The ground-truth session is extracted from the movie *Forrest Gump*. The highlighted text represents some key points of the narrative.

This example provides additional support to the idea of leveraging both kinds of attentive blocks for semantic matching. Furthermore, we can also see that some phrases such as "Gump is Jenny's" and "you are my" have higher similarity after training. This similarity cannot be captured by shallow semantics, but is based on a deeper understanding of the contextual information. This example shows the usefulness of the attention mechanism for information interactions and their aggregation.

6.6 Case Study

We show some results generated by ScriptWriter-CPre, ScriptWriter, and MSN in Figure 12. As we highlighted, there are three key points in the narrative. The ground-truth session is perfectly consistent with the narrative. Both ScriptWriter-CPre and ScriptWriter successfully cover all three key points, whereas MSN misses one point.

Looking at the details in the first two lines, ScriptWriter-CPre chooses the best response probably because it captures the semantic matching between "go home later" in the narrative and "I'll stay" in the dialogue, benefiting from multi-grained representations. In the third line, although ScriptWriter-CPre selects a nonoptimal response, this response does not conflict with either the context or the narrative. Compared with the 3rd line generated by ScriptWriter, the line generated by our method is more related to the given narrative. We attribute this to the design of the narrative representation optimization. Finally, ScriptWriter-CPre selects the response "The best friend" in the last line that matches the remaining key point of the narrative. On the contrary, MSN selects one (the fourth line) that is redundant with the previous utterance (the third line). This example shows that the method we propose in this paper can better serve our primary goal - covering the key aspects of a narrative.

6.7 Error Analysis

Finally, we conduct an error analysis to investigate the cases that our model cannot handle correctly and summarize some research questions for future work. We randomly sample 50 sessions that are different from the ground-truth and categorize their errors into four groups:

(1) The generated lines are inconsistent with the narrative or incoherent with the context (46%). As shown in the first case of Table 6, the second and last lines are irrelevant to the narrative. Even though our method considers both narrative-response and context-response matching, the matching features are aggregated together for response selection. A response candidate can be selected because it highly matches either the narrative or the context, rather than both. The aggregation mechanism may not be sufficient to impose good matches for both. We believe that this problem could be solved by setting a gate for both kinds of matching to guarantee the selected response can match both narrative and context.

Table 6. Examples of typical errors. Four kinds of errors are observed, including (1) being inconsistent with the narrative or incoherent with the context; (2) generating redundant lines; (3) generating information not included in the narrative; and (4) selecting a wrong line for the session ending.

Narrative: Sophie doesn't want to help the Witch, and she ask Howl to go. **Ground-truth Script**: (1) So, try it yourself! (2) I'm not so kind to give you a hand (3) Come on Howl! (4) EOS **Generated Script**: (1) So, try it yourself! (2) Sophie, let's get closer. (3) Come on Howl! (4) I want to eat Howl's heart. **Error**: Inconsistent; No ending
Narrative: Maureen swam back to the bottom of the sea, asking other fish if they saw a boat. But they didn't reply Maureen. **Ground-truth Script**: (1) Do you see a boat? (2) A white boat! (3) They took my son! (4) My son! Help me! Please! (5) EOS **Generated Script**: (1) Do you see a boat? (2) Do you see it? (3) Do you see it? (4) That has nothing to do with you! (5) Talk to me at least! **Error**: Redundant; No ending
Narrative: Father is confessing. **Ground-truth Script**: (1) You are reported in the newspaper, son. (2) They say you do a lot of bad things. (3) They describe how the government force you to commit suicide. (4) When we think about this, maybe we are wrong, but it is always your home. (5) EOS **Generated Script**: (1) You are reported in the newspaper, son. (2) Have you heard it? (3) EOS **Error**: Not covered

(2) The generated lines are redundant (26%). As shown in the second case of Table 6, the second and third lines are the same. This problem has also been addressed by other retrieval-based methods [56]. The reason is that existing matching-based methods pay much attention to semantic matching but neglect to model text coherence. Some researchers have tried to alleviate this problem by involving pre-trained language models [14, 56]. However, this is an unsolved problem and needs further exploration.

(3) The ground-truth lines contain information which is not covered by the narrative (22%), as shown in the third case of Table 6. These cases are very difficult for models to handle since only the context information can be used to select the proper response. Similar to the previous problem, we think modeling text coherence can be a possible way to alleviate this problem.

(4) The model cannot select an ending for a session of scripts (6%), as shown in the first two cases of Table 6. In our experiments, we add a special token "EOS" to mark the "end of session". This token is appended to each session of lines. We find that our model seldom selects this token at the end of a session but tends to select other lines related to the context or narrative. The potential reason is that the model is unable to recognize the ending situation. In other words, the matching model can hardly match a special token with either the context or the narrative. We plan to design an additional mechanism for ScriptWriter-CPre to decide if a session is finished.

7 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 lines in a movie script. We proposed a model that uses the narrative to guide line

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 narrative as an additional context. They also showed the importance of using the narrative in the proper manner.

As a first investigation into 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 the correct order. Other methods to track the dialogue state and the coverage of the narrative can also be designed. We have limited ourselves to retrieval-based script generation. It would be interesting to extend the method to a generation-based approach. Further investigations are thus required to fully understand how narratives can be effectively used in dialogue generation.

ACKNOWLEDGMENTS

This work was supported by Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098 and Shanghai Bilibili Technology Co., Ltd.

REFERENCES

[1] 2020. *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events, NUSE@ACL 2020, Online, July 9, 2020*. Association for Computational Linguistics. https://aclanthology.org/volumes/2020.nuse-1/

[2] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *CoRR* abs/1607.06450 (2016). arXiv:1607.06450 http://arxiv.org/abs/1607.06450

[3] Selmer Bringsjord and David Ferrucci. 1999. *Artificial Intelligence and Literary Creativity: Inside the Mind of Brutus, A Storytelling Machine*. Psychology Press.

[4] Marc Cavazza, Fred Charles, and Steven J. Mead. 2002. Planning characters' behaviour in interactive storytelling. *Comput. Animat. Virtual Worlds* 13, 2 (2002), 121–131. https://doi.org/10.1002/vis.285

[5] 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*. Association for Computational Linguistics, Minneapolis, Minnesota, 1–11. https://doi.org/10.18653/v1/W19-2401

[6] Kenneth Mark Colby. 1975. *Artificial Paranoia: A Computer Simulation of Paranoid Process*. Pergamon Press.

[7] Alexandra DeLucia, Aaron Mueller, Xiang Lisa Li, and João Sedoc. 2020. Decoding Methods for Neural Narrative Generation. *CoRR* abs/2010.07375 (2020). arXiv:2010.07375 https://arxiv.org/abs/2010.07375

[8] Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, 889–898. https://doi.org/10.18653/v1/P18-1082

[9] Angela Fan, Mike Lewis, and Yann N. Dauphin. 2019. Strategies for Structuring Story Generation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 2650–2660. https://doi.org/10.18653/v1/p19-1254

[10] Xiaocheng Feng, Ming Liu, Jiahao Liu, Bing Qin, Yibo Sun, and Ting Liu. 2018. Topic-to-Essay Generation with Neural Networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 4078–4084. https://doi.org/10.24963/ijcai.2018/567

[11] Jianfeng Gao, Michel Galley, and Lihong Li. 2018. Neural Approaches to Conversational AI. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*. ACM, 1371–1374. https://doi.org/10.1145/3209978.3210183

[12] Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. 2005. Story plot generation based on CBR. *Knowl. Based Syst.* 18, 4-5 (2005), 235–242. https://doi.org/10.1016/j.knosys.2004.10.011

[13] 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. AAAI Press, 5110–5117. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16710

[14] Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020. Speaker-Aware BERT for Multi-Turn Response Selection in Retrieval-Based Chatbots. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 2041–2044. https://doi.org/10.1145/3340531.3412330

[15] Jia-Chen Gu, Zhen-Hua Ling, and Quan Liu. 2019. Interactive Matching Network for Multi-Turn Response Selection in Retrieval-Based Chatbots. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. ACM, 2321–2324. https://doi.org/10.1145/3357384.3358140

[16] 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*. AAAI Press, 6473–6480. https://doi.org/10.1609/aaai.v33i01.33016473

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[18] Matthew Henderson, Blaise Thomson, and Steve J. Young. 2014. Word-Based Dialog State Tracking with Recurrent Neural Networks. In *Proceedings of the SIGDIAL 2014 Conference, The 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 18-20 June 2014, Philadelphia, PA, USA*. The Association for Computer Linguistics, 292–299. https://doi.org/10.3115/v1/w14-4340

[19] Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional Neural Network Architectures for Matching Natural Language Sentences. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*. 2042–2050. https://proceedings.neurips.cc/paper/2014/hash/b9d487a30398d42ecff55c228ed5652b-Abstract.html

[20] Harsh Jhamtani and Taylor Berg-Kirkpatrick. 2020. Narrative Text Generation with a Latent Discrete Plan. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020 (Findings of ACL, Vol. EMNLP 2020)*. Association for Computational Linguistics, 3637–3650. https://doi.org/10.18653/v1/2020.findings-emnlp.325

[21] Zongcheng Ji, Zhengdong Lu, and Hang Li. 2014. An Information Retrieval Approach to Short Text Conversation. *CoRR* abs/1408.6988 (2014). arXiv:1408.6988 http://arxiv.org/abs/1408.6988

[22] Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally Coherent Text Generation with Neural Checklist Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*. The Association for Computational Linguistics, 329–339. https://doi.org/10.18653/v1/d16-1032

[23] Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential Latent Knowledge Selection for Knowledge-Grounded Dialogue. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. https://openreview.net/forum?id=Hke0K1HKwr

[24] 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*. http://arxiv.org/abs/1412.6980

[25] 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*. ISCA, 122–125. http://www.isca-speech.org/archive/icslp_2000/i00_2122.html

[26] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. The Association for Computational Linguistics, 110–119. https://doi.org/10.18653/v1/n16-1014

[27] Jiwei Li, Michel Galley, Chris Brockett, Georgios P. Spithourakis, Jianfeng Gao, and William B. Dolan. 2016. A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics. https://doi.org/10.18653/v1/p16-1094

[28] Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to Select Knowledge for Response Generation in Dialog Systems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*. ijcai.org, 5081–5087. https://doi.org/10.24963/ijcai.2019/706

[29] 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 SIGDIAL 2015 Conference, The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 2-4 September 2015, Prague, Czech Republic*. The Association for Computer Linguistics, 285–294. https://doi.org/10.18653/v1/w15-4640

[30] Zhengyi Ma, Zhicheng Dou, Yutao Zhu, Hanxun Zhong, and Ji-Rong Wen. 2021. One Chatbot Per Person: Creating Personalized Chatbots based on Implicit User Profiles. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in*

Information Retrieval, Virtual Event, Canada, July 11-15, 2021. ACM, 555–564. https://doi.org/10.1145/3404835.3462828

[31] 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*. William Kaufmann, 91–98. http://ijcai.org/Proceedings/77-1/Papers/013.pdf

[32] Tomás 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*. http://arxiv.org/abs/1301.3781

[33] Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards Controllable Story Generation. In *Proceedings of the First Workshop on Storytelling*. Association for Computational Linguistics, New Orleans, Louisiana, 43–49. https://doi.org/10.18653/v1/W18-1505

[34] Hongjin Qian, Zhicheng Dou, Yutao Zhu, Yueyuan Ma, and Ji-Rong Wen. 2021. Learning Implicit User Profile for Personalized Retrieval-Based Chatbot. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management, CIKM 2021, Virtual Event, Australia, November 1-5, 2021*. ACM, New York, NY, USA, 1467–1477. https://doi.org/10.1145/3459637.3482269

[35] Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Assigning Personality/Profile to a Chatting Machine for Coherent Conversation Generation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 4279–4285. https://doi.org/10.24963/ijcai.2018/595

[36] 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. (1989).

[37] Mark O. Riedl and Robert Michael Young. 2010. Narrative Planning: Balancing Plot and Character. *J. Artif. Intell. Res.* 39 (2010), 217–268. https://doi.org/10.1613/jair.2989

[38] Alan Ritter, Colin Cherry, and William B. Dolan. 2011. Data-Driven Response Generation in Social Media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 583–593. https://aclanthology.org/D11-1054/

[39] Melissa Roemmele. 2019. Identifying Sensible Lexical Relations in Generated Stories. In *Proceedings of the First Workshop on Narrative Understanding*. Association for Computational Linguistics, Minneapolis, Minnesota, 44–52. https://doi.org/10.18653/v1/W19-2406

[40] 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*. ISCA. http://www.isca-speech.org/archive/icslp_1998/i98_1153.html

[41] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*. AAAI Press, 3776–3784. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11957

[42] Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2017. A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*. AAAI Press, 3295–3301. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14567

[43] 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 of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, 1577–1586. https://doi.org/10.3115/v1/p15-1152

[44] Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric P. Xing, and Zhiting Hu. 2019. Target-Guided Open-Domain Conversation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 5624–5634. https://doi.org/10.18653/v1/p19-1565

[45] Chongyang Tao, Wei Wu, Can Xu, Wenpeng Hu, Dongyan Zhao, and Rui Yan. 2019. One Time of Interaction May Not Be Enough: Go Deep with an Interaction-over-Interaction Network for Response Selection in Dialogues. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 1–11. https://doi.org/10.18653/v1/p19-1001

[46] 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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*. Association for Computational Linguistics, 231–236. https://doi.org/10.18653/v1/P17-2036

[47] Gokhan Tur and Renato De Mori. 2011. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. John Wiley & Sons.

[48] Alan M. Turing. 1990. Computing Machinery and Intelligence. In *The Philosophy of Artificial Intelligence*. Oxford University Press, 40–66.
[49] Scott R Turner. 1994. MINSTREL: A Computer Model of Creativity and Storytelling. (1994).
[50] 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, December 4-9, 2017, Long Beach, CA, USA*. 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html
[51] Oriol Vinyals and Quoc V. Le. 2015. A Neural Conversational Model. *CoRR* abs/1506.05869 (2015). arXiv:1506.05869 http://arxiv.org/abs/1506.05869
[52] 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 (NIST Special Publication, Vol. 500-246)*. National Institute of Standards and Technology (NIST). http://trec.nist.gov/pubs/trec8/papers/qa_report.pdf
[53] 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*. AAAI Press, 2835–2841. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11897
[54] Yeyi Wang, Li Deng, and Alex Acero. 2011. Semantic Frame-based Spoken Language Understanding. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech* (2011), 41–91.
[55] Joseph Weizenbaum. 1966. ELIZA - a computer program for the study of natural language communication between man and machine. *Commun. ACM* 9, 1 (1966), 36–45. https://doi.org/10.1145/365153.365168
[56] Taesun Whang, Dongyub Lee, Dongsuk Oh, Chanhee Lee, Kijong Han, Dong-hun Lee, and Saebyeok Lee. 2021. Do Response Selection Models Really Know What's Next? Utterance Manipulation Strategies for Multi-turn Response Selection. (2021), 14041–14049. https://ojs.aaai.org/index.php/AAAI/article/view/17653
[57] Jason D. Williams and Steve J. Young. 2007. Partially observable Markov decision processes for spoken dialog systems. *Comput. Speech Lang.* 21, 2 (2007), 393–422. https://doi.org/10.1016/j.csl.2006.06.008
[58] Chao-Chung Wu, Ruihua Song, Tetsuya Sakai, Wen-Feng Cheng, Xing Xie, and Shou-De Lin. 2019. Evaluating Image-Inspired Poetry Generation. In *Natural Language Processing and Chinese Computing - 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9-14, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11838)*. Springer, 539–551. https://doi.org/10.1007/978-3-030-32233-5_42
[59] 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 Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics, 3794–3804. https://doi.org/10.18653/v1/p19-1369
[60] Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential Matching Network: A New Architecture for Multi-turn Response Selection in Retrieval-Based Chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Association for Computational Linguistics, 496–505. https://doi.org/10.18653/v1/P17-1046
[61] Yu Wu, Wei Wu, Dejian Yang, Can Xu, and Zhoujun Li. 2018. Neural Response Generation With Dynamic Vocabularies. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (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*. AAAI Press, 5594–5601. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16135
[62] 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*. AAAI Press, 3351–3357. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14563
[63] 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*. AAAI Press, 5610–5617. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16510
[64] Wei Xu and Alexander I. Rudnicky. 2000. Task-based Dialog Management Using an Agenda. In *ANLP-NAACL 2000 Workshop: Conversational Systems*. https://www.aclweb.org/anthology/W00-0309
[65] 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 (2001), 119–139. https://doi.org/10.1080/09528130010029820
[66] 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*. ACM, 55–64. https://doi.org/10.1145/2911451.2911542

[67] 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*. AAAI Press, 7378–7385. https://doi.org/10.1609/aaai.v33i01.33017378

[68] Keen You and Dan Goldwasser. 2020. "where is this relationship going?": Understanding Relationship Trajectories in Narrative Text. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2020, Barcelona, Spain (Online), December 12-13, 2020*. Association for Computational Linguistics, 168–178. https://aclanthology.org/2020.starsem-1.18/

[69] Chunyuan Yuan, Wei Zhou, Mingming Li, Shangwen Lv, Fuqing Zhu, Jizhong Han, and Songlin Hu. 2019. Multi-hop Selector Network for Multi-turn Response Selection in Retrieval-based Chatbots. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics, 111–120. https://doi.org/10.18653/v1/D19-1011

[70] 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, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. Association for Computational Linguistics, 3740–3752. https://aclanthology.org/C18-1317/

[71] Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. 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*. ijcai.org, 4623–4629. https://doi.org/10.24963/ijcai.2018/643

[72] Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. 2018. Multi-Turn Response Selection for Chatbots with Deep Attention Matching Network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics, 1118–1127. https://doi.org/10.18653/v1/P18-1103

[73] 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 (2020), 27–48. https://doi.org/10.1007/s10791-019-09364-x

[74] Yutao Zhu, Jian-Yun Nie, Kun Zhou, Pan Du, and Zhicheng Dou. 2021. Content Selection Network for Document-Grounded Retrieval-Based Chatbots. In *Advances in Information Retrieval - 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 - April 1, 2021, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12656)*. Springer, 755–769. https://doi.org/10.1007/978-3-030-72113-8_50

[75] Yutao Zhu, Jian-Yun Nie, Kun Zhou, Pan Du, Hao Jiang, and Zhicheng Dou. 2021. Proactive Retrieval-based Chatbots based on Relevant Knowledge and Goals. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 2000–2004. https://doi.org/10.1145/3404835.3463011

[76] Yutao Zhu, Ruihua Song, Zhicheng Dou, Jian-Yun Nie, and Jin Zhou. 2020. ScriptWriter: Narrative-Guided Script Generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics, 8647–8657. https://doi.org/10.18653/v1/2020.acl-main.765