Generating Character Lines in Four-Panel Manga

Michimasa Inaba
The University of Electro-Communications
1-5-1, Chofugaoka, Chofu, Tokyo, Japan
m-inaba@uec.ac.jp

Abstract

Automatic content generation based on natural language processing is an active research area, especially for story generation. Research on story generation has focused on generating consistent text pertaining to characters' actions and events; however, there have been few studies on generating characters' lines (utterances) and dialogues. Story plots are not created to stand alone, but are instead used as the basis for the next step in the creation process, such as creating the scripts of movies or plays, the storyboards for comics, and the main body of novels. This paper proposes a Progress-aware and Sample-based Line-Generation Model (PSLGM) that bridges the gap between automatic story generation and practical content generation. The PSLGM estimates the progress of a given plot from context and uses line samples to generate characters' lines based on the plot. Line-generation experiments using a novel dataset created using Japanese four-panel mangas revealed that the PSLGM can generate lines that follow given plots and match characters' personalities.

1 Introduction

Automatic content generation based on natural language processing has attracted considerable attention, particularly for story generation. Early studies involved manually created rules, knowledge-based planning, and case-based reasoning (Meehan, 1977; Lebowitz, 1987; Perez and Sharples, 2001; Gervás et al., 2004; Porteous and Cavazza, 2009; Li et al., 2013).Recently, various neural network-based story-generation models have been proposed that do not require manual preparation of rules or knowledge bases (Fan et al., 2018; Ni et al., 2019; Liu et al., 2020a; Ammanabrolu et al., 2021).

Research on story generation has focused on generating consistent text pertaining to characters' actions and events; however, there are no studies on generating characters' lines (utterances) and dialogues. Existing study has been proposed a method and dataset for selecting an appropriate line to a given plot and context (Zhu et al., 2020). However, no speaker information is provided for the utterances in this dataset, and we cannot distinguish whether successive utterances are by the same speaker or not. This significantly increases the difficulty of line generation, and in actual, no existing work addressed the problem of generating speaker's lines following a given plot and context using this dataset. Story plots do not stand alone, but are used as the basis for the next step in the creation process, such as for creating the scripts of movies or plays, the storyboards for comics, and the main body of novels. Automatic story generation is a challenging task from the perspective of natural language generation. Content generation is limited to the creation of intermediate products in the content.

This paper proposes a model for generating characters' lines based on the story. Herein, we focused on four-panel mangas, one of the simplest content types with a story, which is compatible in genre and story length with story generation datasets such as ROCStories (Mostafazadeh et al., 2016). Table 1 presents an example of line generation based on the manga shown in Figure 1.

In addition to the problem of consistent text generation, which is essential in story generation, two other challenges must be addressed. The first challenge is the generation of lines that follow a given plot. For generating a story from a given title (Fan et al., 2018; Yu et al., 2021), flexible language generation is allowed, although the domain is constrained to a certain extent by the title. A planning-based response generation method for dialogue systems has been proposed, in which there is a one-to-one semantic correspondence between the information (plan) and the utterance to be generated (Nayak et al., 2017). By contrast, our task requires generating lines that follow a given plot,

Story plot (summary)	Chris asks Betty to order a large portion of breakfast as usual. Dorothy criticizes that gluttony will make Chris fat in the middle age, but Chris excuses himself because he has a PE class. Betty defends that he cannot do a good job if he is hungry. Chris is entranced by Betty's kindness.
Context	Betty : Would you order the usual breakfast set? Chris: Yes. I will get an extra-large set. Dorothy: You are a glutton. Chris: Because I have a PE today and I'm starving.
Next speaker	Dorothy
Actual line	I bet you will be fat when you reach middle age.
Generated line	You will get fat when you reach middle age.

Table 1: Example of story, context, next speaker, correct line and generated line in our model. (Originally written in Japanese.)

① Would you order the usual breakfast set?
② Yes. I will get an extra-large set.
③ You are a glutton.
④ Because I have a PE today and I'm starving.
⑤ I bet you will be fat when you reach middle age.
⑥ It is true that the stomach carries the feet.
⑦ Betty, you are so sweet.
⑧ Oh well!

Figure 1: Example of an our four-panel manga we originally created for this study.

and the story and lines do not have a one-to-one correspondence. Additionally, because actions and situations pertaining to multiple speakers are written together in a plot, the models must consider which parts of the plot should be utilized, depending on the context and speaker. The second challenge is to generate lines that match the speaker's personality. The character-centric story generation model (Liu et al., 2020a) represents each character as an embedding vector using distributed representations of verbs associated with the character, and predicts consistent actions using actions (verbs) appropriate for each character. In dialogue systems, various neural response generation models that incorporate speaker information have been proposed (Li et al., 2016; Zheng et al., 2020; Liu et al., 2020b). As a corpus for training these models, the PERSONA-CHAT dataset (Zhang et al., 2018) was created by presenting profile sentences to cloud workers and having them interact with each other according to the given settings. The profile sentences mainly comprise biographical information, objective facts, and hobbies and preferences, such as "I have a dog." and "I like autumn." This personal information is helpful for generating self-disclosing utterances in a chat dialogue between people who have never met before. However, in the generation of characters' lines based on a story plot, the profile sentences are effective only in limited situations. Additionally, since the story plot is not written by each character separately and is written in chronological order, the models cannot freely utilize it for generation. The models must instead recognize the dialogue situation as well as the speaker and then select the parts of the given story plot for line generation.

In this study, we propose a Progress-aware and Sample-based Line Generation Model (PSLGM) that estimates the progress of a given plot from context and uses speaker's line samples to generate characters' lines based on the plot. Additionally, we create a new dataset comprising story plots and dialogue pairs to train and evaluate our model by annotating four-panel mangas.

The contributions of this study are as follows: (1) We propose a story plot-based line-generation task, which follows after story generation. Given a story plot, which is the target of the story generation task, our task is to generate the lines of characters, (2) We propose a PSLGM, a simple yet powerful model for line generation, (3) The

results of dialogue-generation experiments using a dataset created from four-panel mangas indicate that the PSLGM can generate lines that follow given plots and match characters' personalities and outperforms baseline models, including large pre-trained language models.

2 Story-Plot-Based Line Generation

Herein, we propose a model for generating the lines of the next speaker based on a given plot and the history of lines between characters. First, we present the formulation of our story plot-based line generation task and its notation.

The model receives the story plot S, context C and next speaker name U as inputs. Story plot S is a text comprising one to five sentences describing a story. Context C is a history of lines, i.e., text written in the format "(speaker name 1) : (utterance 1), (speaker name 2) : (utterance 2) ... (speaker name N) : (utterance N)." Next speaker's name U is a text of character name whose lines are to be generated.

From the abovementioned input, the model outputs a line Y for the next speaker U. We follow the same generation settings as presented in the previous response generation studies (Li et al., 2016; Zheng et al., 2020), where responses (lines) are generated in an utterance-by-utterance manner.

3 Proposed Model

As shown in Figure 2, the PSLGM contains two encoders, two decoders, a progress estimator and a line database (DB). We introduce all the modules in the following subsections.

3.1 Pretrained Transformer

For the two encoders and decoders in our model (the blue and green blocks, respectively, presented in Figure 2), we use a pretrained transformer architecture, which has been demonstrated to be effective in various NLP tasks, such as generation, classification, and extraction (Vaswani et al., 2017; Devlin et al., 2018; Liu et al., 2019; Raffel et al., 2020). We used the corrupted span prediction task proposed in T5 (Text-To-Text Transfer Transformer) (Raffel et al., 2020) for pre-training.

The parameters of the pretrained transformer encoder and decoder are copied to the two encoders (the blue blocks presented in Figure 2) and decoders (the green blocks presented in Figure 2) in the PSLGM and then trained for the line-generation task.

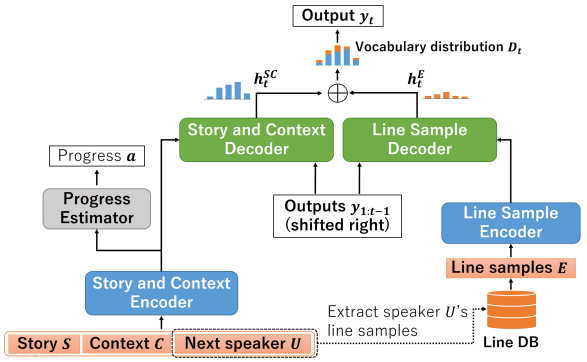

Figure 2: Overview of PSLGM. The blue and green blocks indicate the transformer encoder and decoder, respectively, and the gray block indicates the fully connected layer. The story plot S, context C and speaker U are encoded by the story and context encoder and the encoded output is fed into the progress estimator and story and context decoder. The progress estimator estimates the progress a, which represents how much story plot has been expressed by the context. Then, the line samples E of the speaker U are extracted from the line DB and encoded via the line sample encoder. The encoded line samples are fed into the line sample decoder. The samples are used to reflect the characteristics of speaker U in the lines to be generated. Finally, a weighted sum of the outputs of each decoder is used as the lexical distribution to obtain the output.

3.2 Story and Context Encoder

The input sequence of the story and context encoder is as follows. The story plot S, context C and next speaker U are tokenized by SentencePiece (Kudo and Richardson, 2018) to obtain S' C' and U'. Subsequently, we concatenate S' C' and U' with special tokens ([CLS] and [SEP]) to obtain the input sequence X_{SC} ={[CLS] S' [SEP] C' [SEP] U'}. We feed the input sequence to the transformer encoder, and obtain the distributed representation $R_{SC} = \{r_1^{SC}, ...r_{N_{SC}}^{SC}\}$, where N_{SC} is the length of X_{SC}.

3.3 Progress Estimation Learning

The story plot is described in a chronological order, and the context reflects the story plot from the beginning up to a certain point. In the learning phase, the PSLGM estimates the progress a, which represents how much of the story plot has been expressed in the context. The progress provides a clue to what part of the plot the model should focus on, which can facilitate more appropriate line

generation.

The PSLGM estimates the progress a only in training phase to embed the progress information in the distributed representation of story plot and context R_{SC}. The progress takes values between 0 and 1, and is calculated using r_1^{SC} corresponding to the output of the [CLS] token in R_{SC} as follows.

$$a = \mathbf{sigmoid}(W_a r_1^{SC} + b_a) \quad (1)$$

where W_a and b_a are parameters of the fully connected layer.

Let $\hat{a}$ be the actual progress, we define the progress loss function $\mathcal{L}_{prog}$ using the mean squared error function between a and $\hat{a}$. In experiment, we chose the actual progress $\hat{a}$ = (number of lines in given context C) / (total number of lines in the four panel).

3.4 Line Samples

The generated lines should reflect the characteristics and personality of the speaker. As mentioned in the Introduction, several models have been proposed to reflect the personality of the speaker by providing profile sentences (Li et al., 2016; Zhang et al., 2018; Zheng et al., 2020). However, profile sentences are only useful in limited situations. Therefore, we use line samples collected based on the speaker's personality and characteristics as supplementary information to generate target lines. There are several ways to collect these samples: collect them from utterances by people or characters who are models for the speaker, collect them from another scene of that speaker, or newly create them by hand. In our experiment, we use lines from another scene.

A set of line samples $E = \{e_1, e_2, ..., e_{N_E}\}$ associated with speaker U is obtained from the line DB, where e_i denotes each line sample, while N_E is the number of samples. We used $N_E = 20$ in the experiments. Each sample is tokenized by SentencePiece, prefixed with a [CLS] token, and fed into the line sample encoder. We obtain the vectors corresponding to the [CLS] token from the encoder as the distributed representations of these line samples. Thus, we obtain the set of distributed representations $R_E = \{r_{e_1}, ...r_{e_{N_e}}\}$ corresponding to the line sample set E.

3.5 Line Generation

Finally, we use the output of the story and context decoder using the distributed representation of plot and context R_{SC} and the output of line sample decoder using the distributed representation of line samples R_E for the line generation. The story and context decoder is dedicated for generating appropriate lines related to a given plot and context, whereas the line sample decoder specializes in generating character-specific lines. Our model employs a weighted sum of each decoder's outputs to generate character-specific line that follows the story plot and context.

We use the transformer decoder as the decoders in PSLGM. In the story and context decoder, the output $h_t^{SC} \in \mathbb{R}^{|V|}$ at time t is obtained using token sequence output $y_{1:t-1}$ from time 1 to $t-1$ and the distributed representation of plot and context R_{SC} as inputs. $|V|$ indicates the vocabulary size. Similarly, in the line sample decoder, the output $h_t^E \in \mathbb{R}^{|V|}$ at time t is obtained using token sequence output $y_{1:t-1}$ and the distributed representation of line samples R_E. Notably, the token sequence $y_{1:t-1}$ is the same between the two decoders.

Finally, the vocabulary distribution $D_t \in \mathbb{R}^{|V|}$ at time t is calculated as follows.

$$D_t = \mathbf{softmax}(\alpha h_t^{SC} + (1-\alpha) h_t^E) \quad (2)$$

where the hyperparameter $\alpha = 0.8$ in the experiment.

We define the generation loss function $\mathcal{L}_{gen}$ using the softmax crossentropy function between vocabulary distribution P and actual line Y. The overall model loss function $\mathcal{L}$ is defined on the basis of the generation and progress loss functions $\mathcal{L}_{gen}$ and $\mathcal{L}_{prog}$ as follows:

$$\mathcal{L} = \gamma_{gen}\mathcal{L}_{gen} + \gamma_{prog}\mathcal{L}_{prog} \quad (3)$$

where the hyperparameter γ_{gen} is 1.0 and γ_{prog} is 0.2 in the experiment.

4 Experiment

4.1 Dataset

In the experiment, we created a new dataset using four-panel mangas, one of the simplest content types with a story. Four-panel mangas are also appropriate in size because the story is generally completed in four frames, and the maximum number of lines is approximately 10.

We used two datasets in this experiment. The *inner data* was created using 35 commercially available Japanese four-panel manga titles including genres such as science fiction, fantasy and

	Inner	Open
Book titles	35	4
Four panel mangas	7,686	487
Lines	62,109	3,232
Words per plot	49.59	55.93
Words per line	6.03	6.50

Table 2: Statistics of dataset used in the experiment

ordinary life. The *open data* was created using three titles from the Manga109 dataset (Matsui et al., 2017; Aizawa et al., 2020) (book titles: Akuhamu, OL Lunch and TetsuSan) and a title that we created. The inner data cannot be published owing to copyright considerations; however, the open data (differential data obtained from the three titles in the Manga109 dataset and whole data from our original manga title) is available at https://github.com/1never/MangaLineGeneration/.

For the dataset construction, we transcribed lines in speech bubbles, and created summaries of the four-panel mangas, which were used as the story plots. We hired employees to create summaries of the four panels and transcribe lines presented in speech bubbles. We instructed them to prepare summaries based on the following three criteria: (1) The summary of the four panels should comprise one to five sentences. (2) It should be possible to infer the speaker's lines from the summary by including the names of characters and proper nouns that appear in the lines as often as possible. (3) Lines in bubbles should not be directly included in the summary.

Only text enclosed in speech bubbles was considered as lines for the transcription, and we excluded onomatopoeia and other writing. In the experiment, lines of six characters or less are not used for generation target because some lines are too short (e.g., "Oh") or contain only symbols (e.g., "!!!"),

The statistics of the constructed dataset are presented in Table 2. An example of a four-panel manga is provided in Figure 1, and examples of a summary (story plot) and lines created from Figure 1 are provided in Table 1. Another data example and four-panel manga are shown in Appendix.

4.2 Compared Models

We compared our PSLGM to four existing models and two ablation models to evaluate the effectiveness of our proposed line generation model.

The first model is a vanilla transformer-based seq2seq model (transformer). The number of parameter is the same as those in the PSLGM. It generates a line using a tokenized and concatenated sequence of the story plot, context and next speaker. The second model is the Japanese GPT-2 (GPT-2-Ja). This model has the same architecture as the GPT-2 medium (Radford et al., 2019) and is pretrained using Japanese Wikipedia and Japanese CommonCrawl data (Conneau et al., 2020). We fine-tuned the model with concatenated sequences of a story plot, context, next speaker, and line. For inference, concatenated sequences of the story plot, context, and next speaker are provided, and the lines were generated. The third model is the Japanese Dialogue Transformer (JDT) (Sugiyama et al., 2021), which is a transformer encoder-decoder-based dialogue model with 1.6B parameters trained on 521GB of Japanese Twitter dialogue data. It is one of the most powerful Japanese neural dialogue models among the publicly available ones. For comparison, we use two variations of JDT, i.e., one without fine tuning (JDT w/o ft) and one with fine tuning (JDT), employing the same data as the input and output to the vanilla transformer–based model.

There are two types of ablation models: one without progress estimation learning (w/o progress) and one without line samples (w/o sample).

4.3 Implementation Details

The fairseq toolkit (Ott et al., 2019) was used to implement the PSLGM. For the transformer encoders and decoders used in the PSLGM, the number of layers, hidden layer dimensions, number of attention heads, and vocabulary size are the same as those for the JDT (Sugiyama et al., 2021). The transformer encoders (the story and context encoder and line sample encoder) used in the PSLGM have two layers, and the transformer decoders (story and context decoder and line sample decoder) has 24 layers, with 1,920 dimensions in the hidden layer and 32 attention heads. Vocabulary size of the encoders and decoders is 32K. AdaFactor (Shazeer and Stern, 2018) was used as the optimizer. The learning rate was 5e–5 and the batch size was 48. The optimal learning rate and batch size were determined by changing the combination of hyper-parameters based on validation loss for all models. We trained all models using a single NVIDIA A100 80GB GPU. The PSLGM model took 1 to 2 hours for training and testing.

For pre-training, we used the span prediction task proposed in T5 (Text-To-Text Transfer Transformer) (Raffel et al., 2020). Pre-training was performed from scratch, and the data comprised 200 GB of text data written in Japanese obtained from the Internet, including Wikipedia articles, SNS, forum posts and online novels.

4.4 Experimental Settings

The experiments were conducted under two different settings: the *known* and *unknown* settings. The known setting assumed the generation of sequels and substories with the same characters and worldview as in previous works. Thus, in this setting, the model knew the characters and worldview at the time of inference. We randomly shuffled the inner and open data in this setting and split them into training, development, and test data in a ratio of 8:1:1, respectively.

The unknown setting assumed the generation of new works. In this setting, the model generated the lines of unknown characters with unknown worldviews at the time of inference. We used the inner data as the training data, and the open data were used as the development and test data. We divided the open data into first and second halves for each title, with the first half being the development data and the second half being the test data.

The line DB for the samples was prepared separately for the training, development, and test data. In addition, we did not use lines from the same four panels as samples, which were the target of dialogue generation. For example, when generating a line for speaker A in the second panel of a four-panel manga, the lines of speaker A in the first, third, and fourth panels were excluded from the DB. We used a maximum of 20 samples, and if more than 20 samples were available from the DB, we randomly selected 20 samples. The average number of line samples per input sequence was 11.27 in the experiment.

For all the models, beam search was used for the generation and the beam size was set as 20. To unify the generated line lengths, the minimum generated length and the length penalty were adjusted so that the brabity penalty in BLEU was not less than 0.9.

	BLEU	ROUGE	D-1	D-2
Known setting				
Transformer	0.23	.066	.011	.037
GPT-2-Ja	1.07	.089	.076	.285
JDT w/o ft	1.19	.096	.049	.216
JDT	4.35	.136	**.121**	.401
PSLGM	**8.37**	**.210**	.114	**.422**
w/o progress	8.37	.209	.109	.412
w/o sample	8.15	.205	.105	.403
Unknown setting				
Transformer	0.03	.041	.013	.037
GPT-2-Ja	0.62	.067	.109	.337
JDT w/o ft	1.04	.092	.069	.241
JDT	2.50	.099	**.188**	**.493**
PSLGM	**6.11**	**.173**	.166	.486
w/o progress	5.90	.172	.161	.478
w/o sample	5.84	.170	.159	.479

Table 3: Results of automatic evaluation. Each value represents the average of 10 runs. The known setting (upper) assumes the creation of sequels and sub-stories, and the book titles are shared across the training, development, and test data. The unknown setting (lower) assumes the creation of new works, and the book titles used in the test data are not included in the training and development data.

5 Results and Analysis

5.1 Automatic Evaluation

We used the following automatic measures: BLEU, ROUGE, Distinct-1 (D-1), and Distinct-2 (D-2). BLEU is an algorithm for evaluating machine translation models, while ROUGE is an algorithm for machine summarization models. We used SacreBLEU (Post, 2018) for BLEU implementation. For ROUGE, several variations have been proposed with different sequence matching methods. In our experiment, we used ROUGE-L, which has been reported to correlate highly with human evaluation (Lin and Hovy, 2002), and the implementation in the tf-seq2seq framework.Distinct-[1,2] denotes the number of distinct [1,2]-grams divided by the total number of [1,2]-grams and evaluates the diversity of generated texts.

Table 3 presents the automatic evaluation results. Each value is the average of 10 runs. The results indicate that the PSLGM achieved the best performance in BLEU, ROUGE, and D-2 and the second best in D-1 in the known setting. In the unknown setting, the PSLGM exhibited the best performance in terms of BLEU and ROUGE and the second best

Criterion	Description
Consistency (Cons.)	Is the line in accordance with the given story plot?
Appropriateness (Appr.)	Is the line appropriate to the given context as a response?
Individuality (Ind.)	Does the line reflect the speaker's personality?
Fluency (Flu.)	Is the line grammatically correct?

Table 4: Criteria of manual evaluation

in D-1 and D-2. The D-1 and D-2 measures of the PSLGM were smaller than those of other models in some cases. The D-1 and D-2 results are discussed in section 5.3 along with the subjective evaluation results. Examples of lines generated by each model are shown in Appendix.

5.2 Manual Evaluation

Based on the results of the automatic evaluation, we conducted a manual evaluation using the following four models: JDT, which had the best performance among the compared models, the PSLGM and two ablation models. For reference, we also evaluated the lines actually used in the mangas (oracle).

Lines were evaluated from four aspects: consistency, appropriateness, individuality, and fluency. The details of the criteria are listed in Table 4.

Through the crowdsourcing website Crowdworks (https://crowdworks.jp/), we employed 100 workers in each of the known and unknown settings. The workers were instructed to annotate the generated lines from the four aspects with four possible scores: 1, 2, 3, and 4 (4 indicating the best score). Each worker evaluated the lines generated by each model for 10 pairs of story plots and contexts. Notably, we did not inform the evaluators which lines were the result of which model.

Table 5 presents the results of the manual evaluation. The table also contains the results of the Mann–Whitney U-test between the PSLGM and another model. The results indicate that the PSLGM had the highest scores in all items in both the known and unknown settings. In comparison with the JDT model, significant differences were observed in all criteria. In comparison with the ablation models, significant differences were observed in several criteria except for fluency.

The above automatic and manual evaluation results demonstrate the effectiveness of PSLGM.

	Cons.	Appr.	Ind.	Flu.
Known setting				
JDT	2.34^{*}	2.30^{*}	2.88^{*}	3.35^{+}
PSLGM	**2.58**	**2.63**	**3.12**	**3.38**
w/o progress	2.46^{+}	2.49^{+}	3.12	3.36
w/o sample	2.44^{*}	2.51^{+}	3.09^{+}	3.34
Oracle	3.23	3.24	3.41	3.46
Unknown setting				
JDT	2.32^{*}	2.12^{*}	2.71^{*}	3.18^{*}
PSLGM	**2.70**	**2.71**	**2.89**	**3.31**
w/o progress	2.59^{+}	2.59^{+}	2.86	3.27
w/o sample	2.53^{*}	2.51^{*}	2.80^{+}	3.27
Oracle	3.26	3.24	3.32	3.41

Table 5: Results of manual evaluations. $+$ and $*$ indicate significant differences at the 5% and 1% levels, respectively, between the PSLGM and compared model.

5.3 Analysis

PSLGM vs. JDT Comparing the PSLGM and JDT, the PSLGM showed higher BLEU and ROUGE in the automatic evaluation and significant differences were observed in all items during the manual evaluation. Considering the very poor results of the vanilla transformer model without pretraining (Table 3), pretraining is considered crucial for this task. Therefore, the difference between the PSLGM and JDT is attributed to the task and data for pretraining. JDT is pretrained for the last utterance prediction task using Twitter dialogue data. Because the last utterance in the input sequence is the most important part for the last utterance prediction, the model is trained to emphasize the latter part of the input during the pretraining. In our line generation task, the plot is included in the first part and the context in the remaining part of the input sequence; thus, the story plot is not well utilized in the JDT model. Meanwhile, the PSLGM is pretrained using various data, including the novel text, which is a creative work like manga, and pretrained for the span-prediction task that can only be solved by considering the entire sequences, resulting in natural and appropriate generation.

In the automatic results in the unknown setting in Table 3, the JDT model has the highest D-1 and D-2 scores. Alternatively, the JDT model exhibits the lowest scores in all criteria during the manual evaluation. Therefore, we can conclude that the results of D-1 and D-2 of the JDT model are the outcome of generating several lexically diverse but grammatically and contextually inappropriate lines.

PSLGM vs Ablation models The automatic evaluation results indicate that the PSLGM performed better than the ablation models, indicating that both the progress estimation and the use of line samples are effective.

Comparing the results of manual evaluation between the proposed and w/o progress model, significant differences were observed in consistency and appropriateness in both known and unknown settings, indicating that the progress estimation is effective for natural line generation. Observing the outputs of the w/o progress model, we found that the words included in the given plot are generated in the lines regardless of the context, indicating that progress estimation learning allows the appropriate use of a given plot. Further, we found that using line samples effectively improved consistency, appropriateness, and individuality in both the settings. In particular, the effect of individuality in the known setting was smaller than that in the unknown setting because the characters in the training and test data were often the same. However, there some characters, such as minor characters, that appeared only in the test data; thus, the performance was improved.

5.4 Discussion

To address the issue of reproducibility, we conducted an experiment to examine the data collection policy. We observed changes in the performances of BLEU and ROUGE-L in the unknown setting when the training data and the number of book titles in the training data were decreased from 100% to 10% in 10% increments. In order to ensure an appropriate comparison, the titles to be reduced were manually selected to equalize the amount of training data for the data and title settings. The experimental result is shown in Figure 3. The difference between the data and the title was small for 80% and above, whereas for below 70%, the performance of a smaller number of titles was inferior to that of the data amount. This result suggests that the number of titles is more important than the total amount of data when the amount of data is small.

6 Related Work

The consistency and coherence of the generated lines with respect to the story plot, context, and characters are important for our task. Studies on story generation have addressed consistency and coherence using planning and rule-based approaches (Meehan, 1977; Lebowitz, 1987; Perez and Sharples, 2001; Gervás et al., 2004; Porteous and Cavazza, 2009; Li et al., 2013), as well as neural networks (Fan et al., 2018; Ni et al., 2019; Yu et al., 2021). None of these works, however, focuses on line generation.

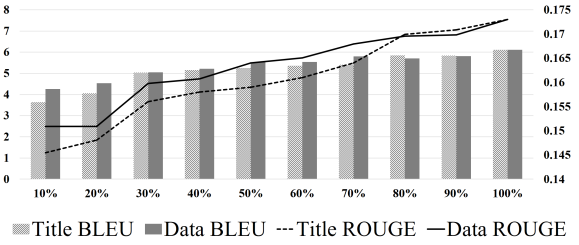

Figure 3: Performance changes when varying the utilization rate of data vs titles for training

GraphMovie dataset (Zhu et al., 2020) is constructed for the task of selecting an appropriate line to a given plot and context. However, no speaker information is provided for the utterances in this dataset. This significantly increases the difficulty of line generation, and in actual, no existing work addressed the problem of generating speaker's lines using this dataset.

Several response generation models have been proposed that use auxiliary sentences. The knowledge-grounded neural conversation model (Ghazvininejad et al., 2018) retrieves sentences related to the input from Wikipedia and feeds them into the model at the same time as the input. In addition, several models have been proposed that retrieve sentences based on semantic relevance to the input sentences and use them for generation (Pandey et al., 2018; Yang et al., 2019; Shen et al., 2020; Wang et al., 2022). In the above studies, the auxiliary sentences must be semantically related to the query. The difference between them and our proposed model is that the latter does not require the samples to be semantically related as long as the lines are from the same speaker.

7 Conclusions

We proposed a line-generation task based on a given plot and context and a model that estimates the progress level and uses line samples to solve this task. The experimental results revealed that our proposed model considerably improved the consistency of the given story plot, the appropriateness for a given context, and the individuality of generated lines compared with those of the baselines.

Our code and proposed model files can be found at https://github.com/1never/MangaLineGeneration/.

Acknowledgements

This research was supported by the NEDO project "Development of Interactive Story-Type Contents Creation Framework."

References

Kiyoharu Aizawa, Azuma Fujimoto, Atsushi Otsubo, Toru Ogawa, Yusuke Matsui, Koki Tsubota, and Hikaru Ikuta. 2020. Building a manga dataset "manga109" with annotations for multimedia applications. *IEEE MultiMedia*, 27(2):8–18.

Prithviraj Ammanabrolu, Wesley Cheung, William Broniec, and Mark O Riedl. 2021. Automated storytelling via causal, commonsense plot ordering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5859–5867.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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

Pablo Gervás, Belén Díaz-Agudo, Federico Peinado, and Raquel Hervás. 2004. Story plot generation based on cbr. In *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 33–46. Springer.

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 AAAI Conference on Artificial Intelligence*, volume 32.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Michael Lebowitz. 1987. Planning stories. In *Proceedings of the 9th annual conference of the cognitive science society*, pages 234–242.

Boyang Li, Stephen Lee-Urban, George Johnston, and Mark O Riedl. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, pages 598–604.

Jiwei Li, Michel Galley, Chris Brockett, Georgios 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 (Volume 1: Long Papers)*, pages 994–1003.

Chin-Yew Lin and Eduard Hovy. 2002. Manual and automatic evaluation of summaries. In *Proceedings of the ACL-02 Workshop on Automatic Summarization-Volume 4*, pages 45–51.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020a. A character-centric neural model for automated story generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1725–1732.

Qian Liu, Yihong Chen, Bei Chen, Jian-Guang Lou, Zixuan Chen, Bin Zhou, and Dongmei Zhang. 2020b. You impress me: Dialogue generation via mutual persona perception. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1417–1427.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2017. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838.

James R Meehan. 1977. Tale-spin, an interactive program that writes stories. In *Proceedings of the 5th international joint conference on Artificial intelligence-Volume 1*, pages 91–98.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849.

Neha Nayak, Dilek Hakkani-Tür, Marilyn A Walker, and Larry P Heck. 2017. To plan or not to plan? discourse planning in slot-value informed sequence to sequence models for language generation. In *INTERSPEECH*, pages 3339–3343.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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)*, pages 188–197.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Gaurav Pandey, Danish Contractor, Vineet Kumar, and Sachindra Joshi. 2018. Exemplar encoder-decoder for neural conversation generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1329–1338.

Rafael Perez and Mike Sharples. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence*, 13(2):119–139.

Julie Porteous and Marc Cavazza. 2009. Controlling narrative generation with planning trajectories: the role of constraints. In *Joint International Conference on Interactive Digital Storytelling*, pages 234–245. Springer.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, Belgium, Brussels. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604.

Siqi Shen, Charles Welch, Rada Mihalcea, and Verónica Pérez-Rosas. 2020. Counseling-style reflection generation using generative pretrained transformers with augmented context. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 10–20.

Hiroaki Sugiyama, Masahiro Mizukami, Tsunehiro Arimoto, Hiromi Narimatsu, Yuya Chiba, Hideharu Nakajima, and Toyomi Meguro. 2021. Empirical analysis of training strategies of transformer-based japanese chit-chat systems. *arXiv preprint arXiv:2109.05217*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. 2022. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pages 3170–3179.

Liu Yang, Junjie Hu, Minghui Qiu, Chen Qu, Jianfeng Gao, W Bruce Croft, Xiaodong Liu, Yelong Shen, and Jingjing Liu. 2019. A hybrid retrieval-generation neural conversation model. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1341–1350.

Meng-Hsuan Yu, Juntao Li, Zhangming Chan, Rui Yan, and Dongyan Zhao. 2021. Content learning with structure-aware writing: A graph-infused dual conditional variational autoencoder for automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6021–6029.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020. A pre-training based personalized dialogue generation model with persona-sparse data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9693–9700.

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*, pages 8647–8657.

A Data Examples

An example of a four-panel manga from Manga109 (Matsui et al., 2017; Aizawa et al., 2020) is provided in Figure 4, and the summary (story plot) created from Figure 4 is provided in Table 8.

Method	Generated Line
Transformer	え？そうなの？ (Huh? Is that so?)
GPT-2-Ja	お腹減ったら戦はできないんだよ (I can't do a good job if I'm hungry.)
JDT w/o ft	ありがとう (Thank you.)
JDT	いつものを大盛りで (I will get an extra-large set.)
PSLGM	中年になったらデブになりますよ (You will get fat when you reach middle age.)
Actual	きっと中年になったらデブよ (I bet you will be fat when you reach middle age.)

Table 6: Examples of generated lines using an example data shown in Table 1.

Method	Generated Line
Transformer	え？え！？ (What? What!?)
GPT-2-Ja	How do I avoid getting fat...? (太らないようにするには...？)
JDT w/o ft	あっそうかっ! (Oh, I got it!)
JDT	あっそうかっ! (Oh, I got it!)
PSLGM	ランチをたくさん食べればいいのよ! (All I have to do is eat so much lunch!)
Actual	アイスが入んないくらいお昼をたくさん食べる！ (I'll eat so much lunch that I can't eat any more ice cream!)

Table 7: Examples of generated lines for the line in the last frame of Figure 4.

Yuko can't resist eating ice cream after lunch. She is worried about getting fat, so she thinks of ways to avoid eating ice cream. Then, she starts to eat more food until she can't eat dessert anymore.

Table 8: Example of a summary created from Figure 4

B Case Study

Table 6 presents lines generated by the compared models and PSLGM using the data displayed in Table 1 as input. Table 7 also presents generated lines for the line in the last frame of Figure 1. As shown in this figure, the lines generated by the PSLGM can follow a given story plot and generate character-specific lines. In addition, the PSLGM can generate various expressions and use the text of the story directly. Conversely, the lines generated by the compared models include lines that are inappropriate with respect to the summary and the context (e.g., Transformer), lines that are analogous to the line in context (e.g., JDT), and lines that are contrary to the plot (e.g., GPT-2-Ja in Table 6).

① Aww, I can't stop eating ice cream after lunch

② You'll get fat!

③ I know

④ How do I...

⑤ Oh, I got it!

⑥ I'll eat so much lunch that I can't eat any more ice cream!

Figure 4: Example of a four-panel manga in the Manga109 (Matsui et al., 2017; Aizawa et al., 2020). (OL Lunch ©Yoko SANRI)