Master's Thesis

Study of Automatic Essay Writing in Japanese

MA Congda

Supervisor Kiyoaki Shirai

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

3, 2021

Abstract

With the rapid development of computer science and technology, mobile devices make it becomes convenient to read electronic texts for relaxation and entertainment. Therefore, to satisfy the desire of people for reading, the demand for electronic texts is increased. Especially, short texts are suitable for reading since they can be read during small pockets of time. However, the number of authors (writers) tends to be decreased. It will cause imbalance between supply and demand of literary work. Text generation is one of the ways which can solve this gap. If texts such as stories and essays can be automatically generated by a computer, the number of texts that people could enjoy would be increased in the future. Text generation is also possible to support the creation of writers by utilizing generated texts as reference material.

Although research on text generation is popular, current Natural Language Generation (NLG) systems are still weak. Although stories can be generated by a neural language model, the lack of coherence harms the quality of them. It also influences the readability of generated texts. Therefore, how to maintain the coherence in the text generation is one of the urgent problems.

In this thesis, we propose a method that automatically generates an essay in Japanese. The essay is a special text with a little limitation in its writing style. The relations between events in an essay are not as close as those in a story or a novel. However, the coherence of sentences in a whole essay is still significant. Two subgoals are set in this thesis. One is to propose a model to automatically generate an essay from a given theme and keywords. The other is to improve the coherence in the automatically generated essays.

At the beginning, we define the essay generation task in this study. One theme and 4 keywords are given by a user to generate an essay. Both the theme and keywords are supposed to be nouns. The whole generated essay is required to describe something about the theme, while the main content of i-th sentence should be coincident with the i-th keyword. In this task, 4 sentences will be generated and be combined into an essay.

To realize the essay generation task, we propose the Essay Generation Model (EGM) that generates sentences of an essay one by one. In the EGM, the previously generated sentence is utilized as the input at the generation of the current sentence. It is inspired by the dialog system, in which an utterance is generated to reply to a user's previous utterance. In this way, the information in the previous sentence can be passed to the next generated

sentence to keep the relevance and coherence between sentences. To implement the EGM, the whole model is decomposed into 2 parts. The first part is used to generate the first sentence of an essay with a theme and a keyword. It is called the First Sentence Generation Model (FSGM). The second part is used to generate the rest of sentences in an essay. It is called the Content Sentence Generation Model (CSGM). In the CSGM, a theme, a keyword, and a previously generated sentence are used as the input, and a sentence is generated as the output. Both two models utilize the sequence-to-sequence model with the attention mechanism and coverage mechanism.

To improve the quality of automatically generated essays with respect to the coherence in them, we propose the Theme-Attention Essay Generation Model (TA-EGM) based on the EGM. The most important difference with the EGM is that the theme is given as the attention in the encoder, not in the input sequence. Through this method, the generated sentences are related to the theme so that the whole essay can be coherent on the theme. The TA-EGM is also decomposed into 2 parts. We call First Sentence Generation Model and Content Sentence Generation Model in the TA-EGM as FSGM-TA and CSGM-TA, respectively.

For training the EGM and TA-EGM, a new dataset is constructed by the essays downloaded from the web site "Aozora Bunko". Essays in "Essay, Review" category under the major category "Japanese Literature" are chosen. Some essays are rather old in "Aozora Bunko". To obtain relatively new essays, only the essays written by the new Japanese character system are downloaded. The number of retrieved essays is 2,140. Before the construction of the datasets for our models, several preprocessing are carried out. Firstly, the old character is replaced by the new character with the "Old and New Kanazukai comparison table". The sentences that do not meet our requirements are removed as well. Secondly, sentences in the essays are split into words by the morphological analyzer MeCab to obtain word sequences used as the input and output of our models. At the same time, the sentences containing more than 78 tokens are removed. Finally, the first noun other than a named entity in the title of an essay is extracted as the theme of the essay. The top 5 keywords extracted from an essay by TF-IDF based scoring are set to the keywords of the essay. From this corpus, two new datasets are constructed. One is a collection of triplets of (theme, keyword, sentence), which is used for training of the FSGM and FSGM-TA. The other is a collection of quadruplet of (theme, previous sentence, keyword, sentence), which is used for training of the CSGM and CSGM-TA.

In the experiment, automatically generated essays are evaluated by human subjects. A blind questionnaire with 4 questions is designed to evaluate the quality of essays. The first three (comprehensibility, relatedness to theme,

and relevance to keyword) evaluate individual sentences in an essay, while the last (coherence in essay) evaluates the overall essay. For every question, the subjects should give a score between 1 to 5. In addition to the EGM and TA-EGM, the baseline model is also evaluated and compared. It produces an essay by generating 4 sentences independently, where a theme and a keyword are given as the input.

From the results, our EGM can generate better coherent essays than the baseline. The human evaluation score on the coherence is increased by 0.12 point. Furthermore, comparing to the baseline model, the improvement by the TA-EGM is 0.31 point. It also outperforms the EGM by 0.19 point. Simultaneously, these three models are similar with respect to the grammatical correctness (comprehensibility). Although the EGM gets a lower score in the relevance between the generated sentences and the theme than the baseline, the TA-EGM where the theme is given as the attention can get comparable scores.

In the future, there are several explorations we should do to revise and improve our model. Firstly, we will search the better essays and explore the better way to extract the theme and keywords from the essay. Secondly, we reconsider what information should be used as the input to improve the quality of generated sentences. Finally, automatic evaluation of essays is essential for the optimization of hyperparameters in the training of the deep learning models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

As one of the most significant research field in Natural Language Processing (NLP), text generation plays a quite important role for realization of sophisticated artificial intelligence. The text generation is a task to generate a text that is as natural as ones written by a human. For example, the text generation technique is used for machine translation when a system outputs a translated sentence for a given source sentence, for automatic summarization when a system produces a summary for a given original document, or for a dialog system when it replies for a user's utterance. In addition, applying text generation technology to automatically create stories and essays has great social significance.

According to the report of the Publishing Association, the publication market of e-books is broadened every year in Japan [7]. At the same time, the number of authors and editors tends to be decreased. Therefore, if stories and essays can be automatically generated by a computer, the number of texts that people could enjoy would be increased in the future. On the other hand, text generation is also possible to support creation of writers by utilizing generated texts as reference material.

Recently, artificial intelligence has rapidly developed in most computer fields. Artificial intelligence becomes similar to a human brain, but it is not exactly the same as human yet. It is still not able to write appropriate description for a thing or an event, even if huge data is supplied for training. Applying statistics, automata, and machine learning, in general, it could just generate simple sentences that simulate a style or format of existing texts. Seeing poor quality of generated texts, the computer seems unable to understand what is important for text generation. As a complicated task

in NLP, text generation involves multiple core areas of NLP. As a result, research themes and problems in the text generation are quite wide. With technological breakthroughs and innovation of a neural network model in NLP, text generation and its applications have also received more attention from researchers.

Although research on text generation is becoming popular, current Natural Language Generation (NLG) systems are still weak. When text generation tasks are required to reach higher levels of creativity, originality, and brevity, the existing text generation systems exhibit limited capabilities. Although stories are possible to be generated by a neural language model, coherence of generated texts is still a problem. A lack of coherence in a story (i.e. sentences are not related each other) heavily lower the quality of it. Therefore, the coherence in word level, sentence level and text level influences the readability of generated texts. How to maintain the coherence in the story generation is one of the most import problems.

To solve this problem, related researchers utilize methods based on a neural network to enhance long memory, such as Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Long Short-Term Memory (LSTM). Recently, the attention mechanism is proposed to weaken the impact of information loss. Even though these methods can be used to control the coherence, generated texts are still not coherent sufficiently especially when they are long. Therefore, research on coherence control in text generation is urgent.

1.2 Goal

The goal of this thesis is to propose a method that automatically generates an essay in Japanese. Here an essay means not a critical report on a certain problem but a text freely written for any specific topics. A human writes anything that she/he has in mind in an essay. The essay is a special text with a little limitation in its writing style. This is quite different from other kinds of literature like a story, novel, poem or haiku in Japanese. Authors could write anything they want to say to others or want to record about a theme. Relations between events in an essay is not as close as those in a story or a novel. However, the coherence of sentences in a whole essay is still important. Unlike most of past studies that have focused on generation of a story or a novel, this study aims at generating an essay.

We assume that a Japanese essay should fulfill several requirements. Firstly, all contents or sentences should be related to a theme. Secondly, every sentence in the essay has a keyword. The role of the keyword in a sen-

tence is like a theme in an essay. It is supposed to represent a core meaning of the sentence. Finally, the description in the essay is freer than other literature. This is important because we do not need to pay much attention on a style or format in the essay generation. However, the generated sentences should be coherent in terms of a theme.

Following the above discussion, two subgoals are set in this research. One is to propose a model to automatically generate an essay from a given theme and keywords. The other is to improve the quality of automatically generated essays with respect to the overall coherence in them.

The proposed method can contribute to provide a wide variety of essays that many people can enjoy. Even when generated essays are not natural and good as ones human writes, they can be used as reference material for essay writers. That is, when they write an essay, they may be able to have some inspiration from automatically generated essays.

1.3 Outline of Thesis

The rest of this thesis is organized as follows. Chapter 2 introduces the related work about several kinds of models in the text generation. Several basic technologies utilized in our models are also presented. Chapter 3 explains details of our two proposed models that can generate coherent essay automatically. Chapter 4 reports results of our experiment to evaluate our proposed methods. Finally, Chapter 5 summarizes the work carried out in this research and discusses future work.

Chapter 2

Related Work

2.1 Text Generation

NLG is a task to generate a new text from an original text, graph, or table. It can be a part of several NLP tasks such as automatic summarization and dialog generation. Although texts are generated from various types of inputs, this thesis aims at generating a text, i.e. an essay, from a given theme. Comparing with other tasks in NLP, text generation is a complicated task. In this section, previous techniques and algorithms associated with the field of text generation are introduced, e.g. storytelling, recipe generation and so on. We borrow some idea from these previous studies to design and implement our system of essay generation in Japanese.

2.1.1 Storytelling

In recent years, storytelling is one of the most popular fields in text generation. From a given theme or keywords, a natural story is automatically generated. The major approaches of the storytelling is roughly shifted with the change of the times as follows: a data-driven approach at an initial stage, statistical knowledge based approach next, and neural network based approach in recent. In every era, researchers try their best to perfect the storytelling task. Nowadays, a general method of generating a story with a neural network consists of two steps. A high-level plan is trained first, then a story is generated from the plan.

McIntyre and Lapata propose a data-driven approach to generate stories [13]. They separate the whole model into three parts: content planning, sentence planning, and surface realization. In the proposed system, it is supposed that a user gives a topic and a length of a story. Firstly, the content planning module decides main entities associated with the given topic and

consults the knowledge base including information about decided entities to get interactions containing the information of subjects and corresponding verbs. To control the coherence in the generated story, time correlation between verbs is also considered. Secondly, in the sentence planning module, a predefined grammar rule is used as a template whose head is a verb. Then arguments of the verb are filled to make a sentence plan. Thirdly, in the surface realization module, sentences are generated by considering detail linguistic constraints such as a tense. A sentence tree, which consists of sentences to be generated as nodes, is constructed. The depth of the sentence tree represents the sentence order in a generated story, and the score of the sentence represents the probability of generating that sentence. Finally, the best story is combined by the language model (tree path) with the highest score.

Fan et al. propose a hierarchical neural story generation model to generate a story from prompts [4]. They apply the two-level processes to realize a planning. First, by the convolutional language model presented by Dauphin et al. [3], they generate a premise or prompt of a story. Then a sketch of the structure of the story is made from the prompt. In their opinion, the prompt makes the generated story easy to remain the global coherence and makes the content richer. Secondly, they use a sequence-to-sequence model to generate a story from the prompt. They apply "Model Fusion" to generate a story more relevant to the prompt. This is an approach that encourages conditioning on the prompt. Their sequence-to-sequence model has access to the hidden states of another pre-trained sequence-to-sequence model. The second model is used to compensate what the first model fails to learn. It also reduces the general problem on training a sequence-to-sequence model, which degenerates into a language model that captures primarily syntactic and grammatical information.

Zhai et al. propose a story generation model that generates stories about daily activities with the coherence [27]. In this model, a symbolic "Agenda Generator" is used to perform text planning. It generates an agenda from a given specific scenario according the temporal graph [25] gotten from the dataset InScript [15]. Then, a neural surface realization module is used to generate words as well as to determine the end of the current event and move to the next event. According to this model, they also illustrate the possibility of guiding the direction of story generation by associating text generation to a latent intention variable.

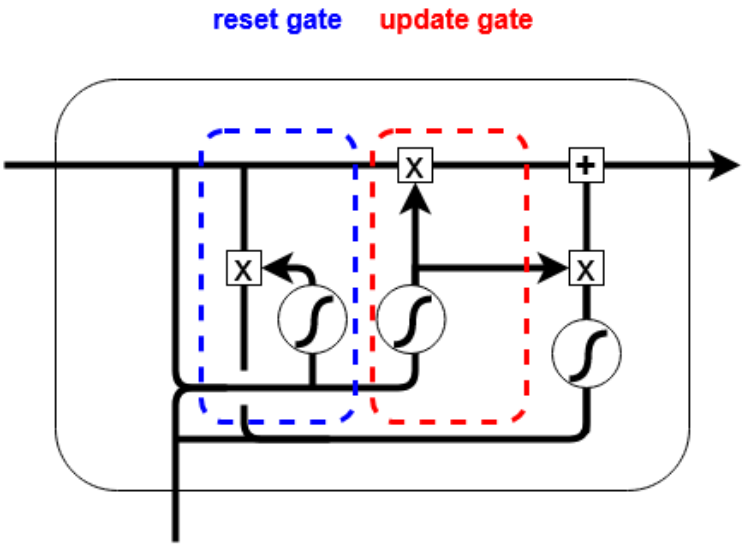

Figure 2.1: Gated Recurrent Unit (GRU) cell

2.1.2 Generation of Other Kinds of Texts

In recent years, text generation becomes widely applied to automatically produce various kinds of texts in different writing styles. For instance, Liu et al. generate Wikipedia articles [9] and Zhang and Lapata generate poetry [29]. Jain et al. add independent descriptions into a short story [6]. Martin et al. use a two-step model to generate a story [12]. First they transfer a text into events, then generate stories as sequences of the events. Harrison et al. use the Markov Chain Monte Carlo (MCMC) sampling technique to generate stories based on movie plots taken from Wikipedia [5]. Kiddon et al. use the "Neural Checklist" model to keep track of the progress of cooking by using ingredient words to generate cooking recipes [8]. Peng et al. control the ending valence and generate endings with different sentiments [16]. Yoneda et al. generate Japanese haiku by a neural network [26].

2.2 Gated Recurrent Unit

In our proposed models, GRU is used as the basic neural network module, which is proposed by Cho et al. [2]. It is one of the recurrent neural networks that accepts a sequence and classify it into certain categories. GRU aims to solve the problem of vanishing gradient which comes with a standard recurrent neural network.

As shown in Figure 2.1, GRU has two gates, update gate and reset gate. They are two vectors that decide what information should be passed to the output. An important role of these gates is to keep information from a long ago without washing it through time, and to remove information that is irrelevant to the classification. In Figure 2.1, the horizontal line in the top represents a memory cell that keeps the information long time. Note that the update gate in red adds information to the memory cell, while the reset gate in blue subtracts information from it.

Equation (2.1) to (2.4) show how the parameters in GRU are updated. First, Equation (2.1) shows the parameter estimation in the update gate. When plugging the token x_t into the GRU, it is multiplied by its weight W^z. The same procedure is applied for h_{t-1} which holds the information of the previous $t-1$ units and is multiplied by its own weight U^z. Both results are added together with the bias b_z. Finally, a sigmoid activation function is applied to squash the result between 0 and 1.

Equation (2.2) shows the parameter estimation in the reset gate. r_t is determined in the same way as the update gate. We plug in h_{t-1} and x_t, multiply them with their corresponding weights, sum the results, and apply the sigmoid function. However, the signals from the reset gate is used differently from the input gate.

In GRU, h'_t is a memory cell, which uses the reset gate to store the relevant information from the past. Equation (2.3) shows how h'_t is calculated. The input x_t is multiplied with a weight W and h_{t-1} with a weight U. Then, we calculate the Hadamard(element-wise) product between the reset gate r_t and Uh_{t-1}, and sum up them with the bias b. Finally, the nonlinear activation function tanh is applied to get the h'_t.

At last, the hidden state for the current unit h_t is calculated as Equation (2.4). The update gate is necessary in this step to determine what to collect from h'_t and h_{t-1}.

$$z_t = \sigma(W^z x_t + U^z h_{t-1} + b_z) \tag{2.1}$$

$$r_t = \sigma(W^r x_t + U^r h_{t-1} + b_r) \tag{2.2}$$

$$h'_t = \tanh(W x_t + r_t \odot U h_{t-1} + b) \tag{2.3}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t \tag{2.4}$$

2.3 Sequence-to-sequence Model

Since our proposed essay generation model is a kind of a sequence-to-sequence model, we introduce the basic concept of it in Subsection 2.3.1. Two additional techniques of the sequence-to-sequence model are also introduced in the rest of subsections.

2.3.1 Encoder and Decoder

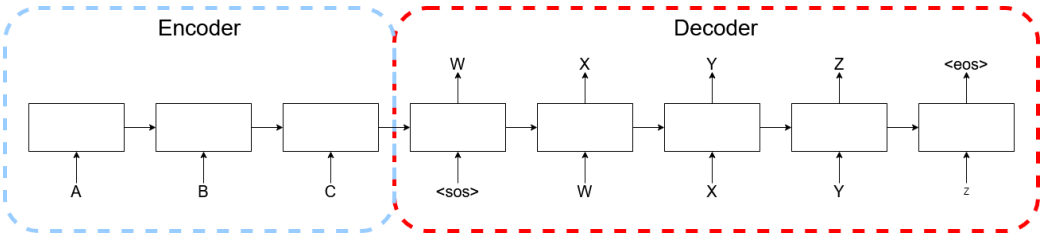

Figure 2.2: Sequence-to-sequence model

Sequence-to-sequence model, also known as Encoder-Decoder model, is a model to convert a sequence to another sequence. It is widely applicable for many NLP tasks, such as machine translation (from a sentence in a source language to a sentence in a target language), dialog system (from user's utterance to system's response), and so on. It is proposed by Sutskever et al. [22]. Sequence-to-sequence neural networks achieved state of the art performance on various tasks in NLP, for example, in machine translation [22] and summarization [18]. Recently, several open-ended generation systems have developed based on this model as well.

The sequence-to-sequence model contains an encoder module and a decoder module as shown in Figure 2.2. Both the encoder and the decoder are implemented by RNN, LSTM or GRU models. In the encoder, the input sequence is transferred into hidden states. The hidden state at the final time step is supposed to include all information of elements of the input sequence. In the decoder, the initial hidden state is the final hidden state of the encoder. The decoder starts generating the output sequence from the initial hidden state. At the same time, the output is also fed into the next timestep for predicting the future outputs. When the special symbol $\langle eos \rangle$ is generated, the decoder stop generating the output sequence.

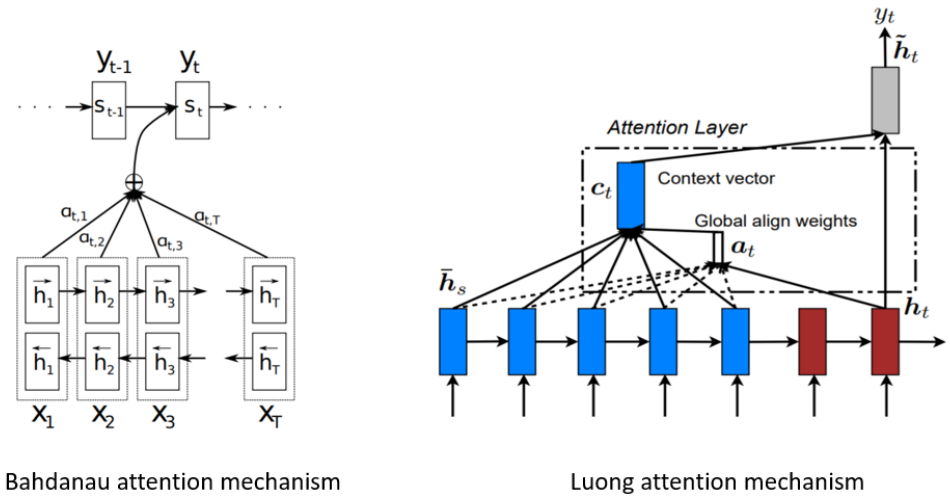

Figure 2.3: Bahdanau Attention [1] and Luong Attention [11]

2.3.2 Attention Mechanism

One of the disadvantages of the sequence-to-sequence model is that the decoder relies too much on the vector that compresses the entire sequence of the input. The entered sequence may contain lots of words, but information about individual words may not properly transferred to the decoder since only the hidden state of the final timestep in the encoder is passed to the decoder. The attention mechanism is a method to solve this problem. The hidden states of every input sequence are kept in the attention layer resulting in reduction of the information loss. Then, we create a unique mapping between each timestep of the decoder to all the encoder hidden states. This means that the decoder can access to the entire input sequence and even can selectively focus on specific elements from that sequence to produce the output instead of just using the last hidden state of the encoder.

Since not all words in the encoder have the same contribution when generating words in the decoder, the attention mechanism gives higher weights to words with important characteristics. There are two different major types of attention mechanisms: Bahdanau Attention [1] and Luong Attention [11]. Figure 2.3 shows an overview of these two models.

The attention mechanism consists of three parts, an align layer, attention weights, and context vector. In Bahdanau Attention, in the align layer, the alignment score evaluates how similar the input data at position t and the output at position i are. The score is calculated by the previous hidden state of the decoder s_{i-1}, output of the decoder y_{i-1} and the hidden state h_t of the

input sequence. To get the attention weight, a softmax activation function is applied to the alignment scores. The context vector c_i is used to compute the final output of the decoder. It is the sum of the attention weights and the hidden states of the encoder, which represents the information of the input sequence specific to the output at time i.

Luong Attention is another attention mechanism. There are 2 differences between Bahdanau Attention and Luong Attention. First, the calculation flow in Bahdanau Attention can be denoted as $s_{i-1} \rightarrow a_i \rightarrow c_i \rightarrow s_i$, where the previous hidden state s_{i-1} is used to calculate the s_i (a_i and c_i means the attention and context vector respectively). But in Luong Attention, the process is $s_i \rightarrow a_i \rightarrow c_i \rightarrow \widetilde{s_i}$, where the current hidden state s_i is used. Second, Bahdanau Attention was only tested on the concat alignment function to form the context vector. However, Luong Attention was tested in several alignment functions. Equation (2.5) shows the three alignment functions designed in Luong Attention, where W_a is a weight matrix to be trained.

$$\text{score}(h_t, s_i) = \begin{cases} h_t^\top s_i & dot \\ h_t^\top W_a s_i & general \\ v_a \top \tanh(W_a[h_t; s_i]) & concat \end{cases} \tag{2.5}$$

2.3.3 Coverage Mechanism

As a common problem of the sequence-to-sequence models, it is known that the same words tend to be generated many times in an output sequence [24, 14, 19, 23]. Such repetition of words drastically declines the quality of generated sentences. Especially, it becomes much serious when a sequence-to-sequence model generates multiple sentences. The coverage mechanism proposed by Tu et al. is a method to alleviate this problem [24]. They use this approach to improve the translation quality and alignment quality in machine translation over the standard attention mechanism.

The coverage mechanism is improved by See et al. [20], who introduce a simpler coverage vector c_i that is the sum of attention distributions over all previous decoder timesteps as shown in Equation (2.6). It evaluates how the words generated until the current timestep are overlapped using the history of the attention.

$$c^i = \sum_{i'=0}^{i-1} a_{i'} \tag{2.6}$$

As an additional input, the coverage vector is added to the attention

mechanism when calculating the attention distribution as shown in Equation (2.7).

$$e_j^i = v^\top \tanh(W_a^\top h_j + W_b^\top s_i + w_c c_i^j + b_{attn}) \tag{2.7}$$

W_a, W_b, w_c and b_{attn} are parameters to be trained. It enables us to determine the current output by considering previous outputs to avoid repetition.

In order to further suppress the repetition, a coverage loss is defined as Equation (2.8). Then, as shown in Equation (2.9), the coverage loss is weighted by a hyperparameter λ and added to the primary loss function to get a new loss function. The coverage loss penalizes the repetition of the same word generated before the timestep i

$$\text{covloss}_i = \sum\nolimits_j \min(a_j^i, c_j^i) \tag{2.8}$$

$$\text{loss}_i = \text{loss(normal)}_i + \lambda \ \text{covloss}_i \tag{2.9}$$

2.4 Originality and Significance

Different from other tasks in the field of text generation, we propose a new task of the essay generation in Japanese. Comparing with other literature, an essay is written in a relatively freer writing style. If essays can be automatically generated by a computer, people can read them for relaxation or fun. On the other hand, the automatic essay generation can help creative activity of essay writers. Since essays can be written freer than stories or novels, there exists a wide variety of essays for a specific topic. Writers can be inspired from many automatically generated essays. Also, the description in automatically generated essays would help authors save energy in thinking detailed portrayal, so they could pay more attention on the plot plan.

In this study, we mainly tackle two problems to generate an essay. One is how to generate an essay related to a given topic, the other is how to improve the coherence in an overall generated essay.

As for the first problem, we propose a method that utilizes a theme and keywords to generate an essay automatically. Different from the general model in storytelling, the plot plan is not designed at the beginning. The essay is generated from not a plan but a given theme and several keywords by our model. The theme is used to determine the content of the essay, and the keyword is used to control the content of each sentence. In this way, we realize the essay generation so that the output text is related to the given and desired theme.

As for the second problem, two generation models are proposed to improve the global coherence. That is, the models are designed to generate coherent sentences that are related to each other, rather than independent ones. In our models, the previously generated sentence is entered as an input at the generation of the current sentence to improve the coherence of two succeeding sentences. Furthermore, in one of our models, the theme is constantly given for generation of all sentences using the attention mechanism. It is expected to improve the coherence of all sentences in the generated essay.

Chapter 3

Proposed Model

3.1 Overview

In this section, the overview of our essay generation model is introduced.

According to previous research on other text generation, the sequence-to-sequence model shows relatively stable results. At the same time, it could handle Japanese words. Therefore, the sequence-to-sequence model is chosen as the basic model to realize automatic generation of essays in Japanese.

The definition of the essay generation task in this study is as follows. $X = [th; k_1, k_2, k_3, k_4]$ is the input of our model. th represents the theme of the essay. The whole generated essay is required to describe something about the theme. "Spring", "football", "music" are examples of the theme of the essay. On the other hand, k_i represents the keyword, which represents the main content of the i-th sentence in the essay. The theme and keywords are given by a user without any restriction, but they must be nouns in Japanese. The task of the essay generation is defined to obtain an essay $t = [t_1, t_2, t_3, t_4]$ for a given X. t_i represents the i-th generated sentence that corresponds to the keyword k_i. In our task, the length of the essay is limited to 4 sentences. Our model generates sentences one by one, then 4 generated sentences are combined into an essay.

In the early stage of this study, an essay generation system is designed to generate a sentence where only a theme and a keyword are given. However, we observed that generated essays were rather poor. Three major problems were (1) the same word was generated many times in one sentence, (2) the generated sentences were not coherent, and (3) the length of the sentence was short. Inspired by generation in a dialog system, we decide to use the previously generated sentence as the input at the generation of the current sentence. That is, a sentence generated by the model is passed to the model

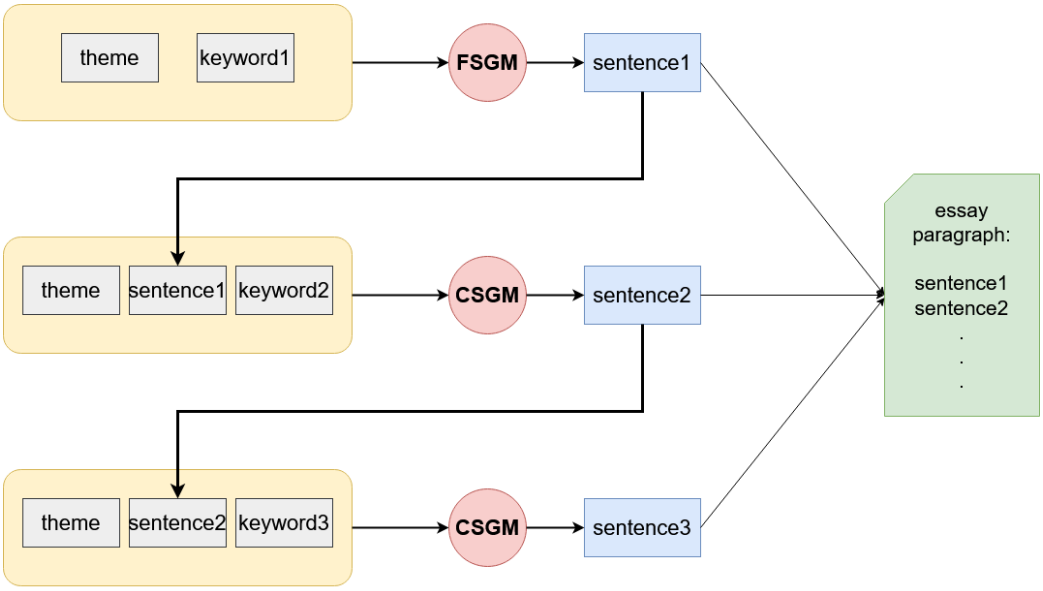

Figure 3.1: The flow chart of proposed model

when it generates the next sentence, similar to a dialog system that generates an utterance to reply a user's previous utterance. In this way, the information in the previous sentence can be passed to the next generated sentence to keep the relevance and coherence between sentences.

Figure 3.1 illustrates how a Japanese essay is generated in our proposed framework. The whole model is decomposed into 2 parts. The first part is used to generate the first sentence. It is called First Sentence Generation Model (FSGM). In this model, the input only contains a theme and a keyword. The second part is called Content Sentence Generation Model (CSGM). This model uses a theme, a keyword, and a previously generated sentence as the input, and generates a sentence as the output. In both two models, the sequence-to-sequence model with the attention mechanism and coverage mechanism is utilized. Although the length (the number of the sentences) of the essay is 4 in our task, Figure 3.1 implies that our model can generate arbitrary number of sentences by applying CSGM repeatedly.

3.2 Essay Generation Model

In this section, the details of our model to generate an essay are presented. Hereafter, we call our model Essay Generation Model (EGM).

3.2.1 Framework

As already explained, the EGM consists of two sub-models: the FSGM and CSGM. Since each is the sequence-to-sequence model, the input of the FSGM and CSGM should be a sequence.

In the FSGM, the theme is paired with the first keyword to form the input data for the encoder. Between the theme and the keyword, a special symbol $[sep]$ is added to distinguish them. The input sequence for the FSGM is represented as Equation (3.1).

$$[th \quad [sep] \quad k_1] \tag{3.1}$$

In the CSGM, the previously generated sentence is added into the input data of the next sentence generation step. As the human's normal way of thinking, theme, and the previous sentence is the premise, while the keyword is provided to control the content of the sentence to be generated next. Therefore the input data is organized in the order of theme, the previous sentence, and the keyword. Also, $[sep]$ is used as the boundaries of them. The input sequence for the CSGM is represented as Equation (3.2).

$$[th \quad [sep] \quad w_1^p \cdots w_n^p \quad [sep] \quad k_i] \tag{3.2}$$

th is a theme, k_i is a keyword of i-th sentence, and $w_1^p \cdots w_n^p$ is a sequence of words of the previous sentence generated by the model.

In both the FSGM and CSGM, the output sequence is a sentence, i.e. a sequence of words or tokens. Two special symbols are used: $\langle sos \rangle$ and $\langle eos \rangle$. $\langle sos \rangle$ represents the beginning (start) of the sentence. It is always given to the decoder as the first token. $\langle eos \rangle$ represents the end of the sentence. The models stop generating tokens when they output $\langle eos \rangle$ or the number of the output tokens reaches a predefined maximun length.

The whole framework of the FSGM and CSGM is presented in Figure 3.2. Recall that the FSGM and CSGM are the sequence-to-sequence models consisting of the encoder and decoder. The module on the left in red is the encoder, while the module on the right in blue is the decoder. The input data of the FSGM and CSGM is defined as $X = [x_1, x_2, \ldots, x_t]$, where x_t represents a token in the input data. x_t can be a theme, a keyword, a word in a previous sentence or the special token $[sep]$. Each token is represented as a one-hot vector, then it is converted to word embedding of 512 dimensions. Initially, the word embedding is randomly set, then it is updated through the training of the sequence-to-sequence model. Then, a hidden state h_t of each word is obtained by GRU as $h_t = \mathrm{GRU}(h_{h_1}, x_t)$.

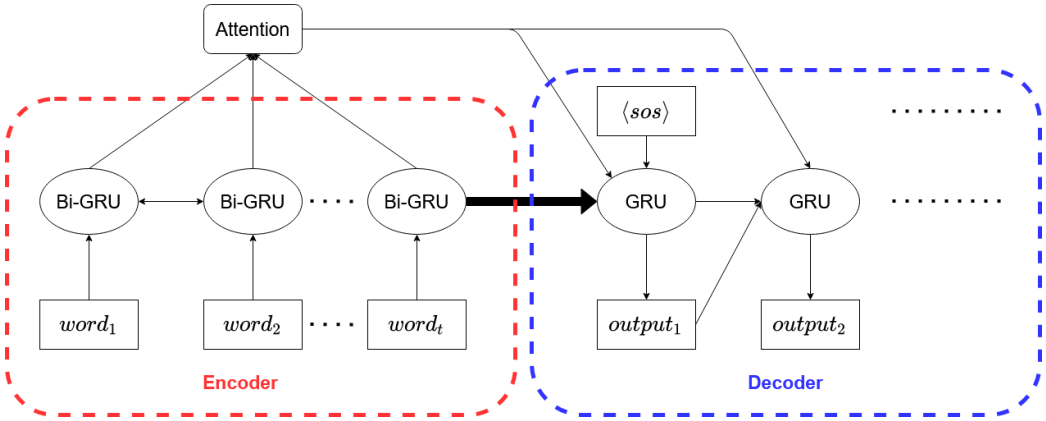

Figure 3.2: Sequence-to-sequence model of FSGM and CSGM

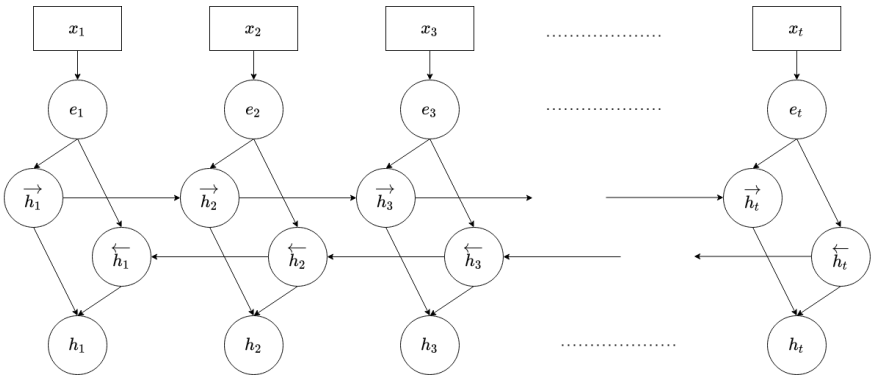

Figure 3.3: Framework of Bi-GRU

In the encoder of the FSGM and CSGM, Bi-directional GRU (denoted as "Bi-GRU" in Figure 3.2) is used. Figure 3.3 shows an overview of Bi-GRU. x_i, e_i, and h_i are the token embedding (one-hot vector), word embedding, and the vector of the hidden state, respectively. Bi-GRU contains a forward layer and a backward layer as shown in Equation (3.3).

$$\begin{aligned} \overrightarrow{h_t} &= \mathrm{GRU}(\overrightarrow{h_{t-1}}, x_t) \\ \overleftarrow{h_t} &= \mathrm{GRU}(\overleftarrow{h_{t-1}}, x_t) \end{aligned} \tag{3.3}$$

The vector $\overrightarrow{h_t}$ contains the semantic information from the head to the current state, while the vector $\overleftarrow{h_t}$ contains the semantic information from the current state to the end of input. These two vectors are concatenated to get the hidden state $h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}]$. In this way, Bi-GRU can consider both left and right contexts when it encodes abstract representation of words.

In the decoder, GRU is applied to generate the sentence word by word. Equation (3.4) shows how y_i, which represents the word generated at position i, is obtained. Here, s_{i-1} represents the hidden state gotten from the timestep $i-1$ in the decoder.

$$y_i = \mathrm{GRU}(y_{i-1}, s_{i-1}) \tag{3.4}$$

In the early stage of this study, instead of GRU, LSTM was also used as the base component of the encoder and the decoder. Through rough comparison between them, we found that GRU could generate the sentences with the same quality as LSTM. In addition, the training time of GRU was faster than LSTM. Since GRU was more efficient, we finally selected GRU for the implementation of the FSGM and CSGM.

Attention Mechanism

In addition to an ordinary sequence-to-sequence model using GRU, an attention mechanism is introduced. When a system generates long sentences, they tend not to be very natural. One of the reason is that the model is not able to encode the input sentence appropriately when the input is long. In our EGM, the input can be long sequence because the previous sentence is given. It is a general problem in the RNN model, which is known as the relevance loss. The attention mechanism can alleviate this problem well. Therefore, it is introduced into our model. A normal global attention mechanism is used in our EGM.

As introduced in Subsection 2.3.2, the attention mechanism consists of three components: the align score, the attention weight, and the context vector. In our model, the align score between s_i and h_j is calculated as Equation (3.5), where s_i is the hidden state in the decoder, which is derived from the previous output word y_{i-1}, and h_j is the hidden state in the encoder.

$$\mathrm{score}(s_i,\, h_j) \;=\; v^\top \tanh(W_a^\top h_j + W_b^\top s_i + b_{attn}) \tag{3.5}$$

v, W_a, W_b, b_{attn} are training parameters. Then, the attention weight α_{ij} is derived by normalization as shown in Equation (3.6):

$$\begin{aligned} \alpha_{ij} \;&=\; \mathrm{score}(s_i,\, h_j) \\ &=\; \frac{\exp(\mathrm{score}(s_i,\, h_j))}{\sum_t \exp(\mathrm{score}(s_i,\, h_j))} \end{aligned} \tag{3.6}$$

From the attention weight, the context vector c_i is gotten as Equation (3.7). It is the weighted sum of the hidden states in the encoder (h_1, h_2, $\cdots$, h_t), where α_{ij} are used as the weights.

$$c_i = \sum_{j=1}^{t} \alpha_{ij} h_j \tag{3.7}$$

Then, using the hidden state s_i and the context vector c_t, a simple concatenation layer is employed to combine them to produce the attentional hidden state $\widetilde{s_i}$ as shown in Equation (3.8). W_c is the weight matrix to be trained. Intuitively, $\widetilde{s_i}$ contains the information of all inputs in the encoder.

$$\widetilde{s_i} \;=\; W_c^{\top}[c_i;\, s_i] \tag{3.8}$$

The attentional vector $\widetilde{s_i}$ is then fed through the softmax layer to produce the predictive distribution formulated as Equation (3.9). W_x is a training parameter. y_i is the final output that represents a word to be generated.

$$p(y_i|y_{<t},\, X) \;=\; \text{softmax}(W_x \widetilde{s_i}) \tag{3.9}$$

Coverage Mechanism

To deal with another problem in generation, repetition of the same word in a generated sentence, the coverage mechanism is imported into our model. It introduces a coverage vector when computing the distribution of attention. The coverage vector is the sum of the attention distributions from data. In every loss calculation step, the coverage loss is added to the loss function. By this procedure, the penalty is given if the attention is placed in the same place of the input data, and the repetition in output sentences could be well suppressed. Firstly, the attention distributions over all previous decoder timesteps are summed up to get the coverage vector c_i.

$$c_i \quad = \quad \sum_{i'=0}^{i-1} a_{i'} \tag{3.10}$$

c_i is a distribution that represents the degree of coverage. It includes the information on words that have been received from the attention mechanism so far. Certainly, the c_0 is a zero vector, since none of the input data has been covered at the first timestep. Then, the calculation of the align score in the attention mechanism is changed from Equation (3.5) to (3.11) by adding $w_c c_i$.

$$\text{score}(s_i,\, h_j) \;=\; v^{\top} \tanh(W_a^{\top} h_j + W_b^{\top} s_i + w_c c_i + b_{attn}) \tag{3.11}$$

w_c is a training parameter vector of the same length as v. It makes the attention mechanism prevent repeatedly paying attention to the same location in the encoder. As a result, the repetition in the generated sentence is restrained.

Secondly, a new loss called coverage loss is defined as follows:

$$covloss_i \quad = \quad \sum\nolimits_i \min(a_i,\, c_i) \tag{3.12}$$

The coverage loss is less than 1. As See et al. [20] considered, the coverage loss of Equation (3.12) differs from the coverage loss used in machine translation. It is more flexible. We only penalize the overlap between each attention distribution and the coverage so far. Just used to prevent repeated attention, with no roughly one-to-one translation ratio. It is a similar effect as in summarization. Since See et al. [20] proposed the coverage mechanism for summarization, no change is made on their coverage mechanism. Finally, a new loss function is redefined as Equation (3.13). It is the sum of the primary loss function (the negative log-likelihood loss) and the coverage loss with the weight hyperparameter λ. We set $\lambda = 1$ in this study.

$$\text{loss}_i \quad = \quad -\log P(w_i^*) + \lambda \sum\nolimits_i \min(a_i,\, c_i) \tag{3.13}$$

3.2.2 Dataset

A collection of (theme, keyword, sentence) is required to train the FSGM, while (theme, previous sentence, keyword, sentence) is required to train the CSGM. This subsection explains how to construct a training data to train our essay generation models.

In the beginning, we try to find the dataset that is suitable for our experiment. However, Japanese datasets in text generation are less than expected. Furthermore, the datasets related to essay generation are absolutely none. Therefore, a new dataset is constructed by ourselves.

First, we search an online collection of essays. We suppose that essays used in the dataset should fulfill the following requirement. First, all contents in the essay should be related to a certain theme. Second, the length of the essay is not too long. It is expected that too long essay is very hard to be automatically generated. We start to generate short essays that may be relatively easy task. Finally, "Aozora Bunko"[1] is chosen as a source of essays. In this web site, copyright free novels and other documents are collected. Among the categories of "Aozora Bunko", we find documents in "Review,

[1] http://yozora.main.jp/9/1/ndc914.html

Table 3.1: Examples of replacement of old characters

old writing	new writing	text
ゑ	え	...未練な私が輪廻ゆ*ゑ*... ...未練な私が輪廻ゆ*え*...
ゆふ	ゆう	...*ゆふ*べの寝まきながら... ...*ゆう*べの寝まきながら...
ゐ	い	...て*ゐ*るが... ...て*い*るが...

Table 3.2: Example of inappropriate replacement of old characters

old writing	new writing	text
を	お	家*を*出て椎の若葉におおわれた... 家*お*出て椎の若葉におおわれた...
けふ	きょう	...だ*けふ*くらんでいて... ...だ*きょう*くらんでいて...
まう	もう	...すぎ去ってし*まう*かもしれない。 ...すぎ去ってし*もう*かもしれない。

Essay" category under the major category "Japanese Literature" can meet our requirements.

The essays in this category are a bit older than we excepted. There are essays written in 2 writing styles: the new Japanese character system ("sinjisinkana" (新字新仮名)) and the old Japanese character system ("kyujisinkana" (旧字新仮名)). Only the essays written by the new Japanese character system are downloaded to ensure the essays are easily readable, contain fewer archaic words, and are closer to modern essays. The number of the essays obtained from "Aozora Bunko" is 2,140.

Preprocessing

After the download of the essays, some preprocessing is carried out for the construction of the dataset. Although the newer writing style essays are only downloaded, there is a considerable number of words described by old way of writing. It can be a problem for training the EGM, since the use of both old and new writing causes the increase of the size of the vocabulary. The "Old and New Kanazukai comparison table" [2] in the official website of the Agency for Cultural Affairs is used to fix this problem. This table compiles

[2]https://www.bunka.go.jp/kokugo_nihongo/sisaku/joho/joho/kakuki/syusen/tosin01/19.html

old characters and their corresponding new characters. Old characters in the essays are replaced with new characters by looking up the table. Table 3.1 shows examples of the replacement. It includes old writing, new writing, original text, and replaced text. However, the replacement is not always appropriate as the examples in Table 3.2. Obviously, the words and sentences after the replacement do not make a sense. However, since such errors are not so often found, they are just ignored.

In a text in Japanese, parentheses are used to show some additional information. Even when phrases in parentheses are removed, a sentence is grammatical and a central meaning of it is not changed. So, all phrases surrounded parentheses in the essays are removed. Here is an example:

(original)　自分は、大川端 (おおかわばた) に近い町に生まれた。
(processed)　自分は、大川端に近い町に生まれた。

The word "おおかわばた" in parentheses means pronunciation of the previous word 大川端 (okawabata) that is a location name. Since it is added as annotation, it can be removed. We find that sentences in an essay are sometimes incomplete and ungrammatical. Here is an example:

僕なんぞもいつ死ぬかわからないが、...

In this sentence, "... " means the rest of the sentence is omitted. Therefore, sentences including symbols (such as "... ") except for the punctuation "、" (comma) and "。" (period) are removed. Note that the number of sentences removed by this preprocessing is low.

Split of Words and Sentences

After the above preprocessing, sentences in the essays are split into words by the morphological analyzer MeCab. The obtained word sequences are used as the input and output of our sequence-to-sequence models.

At first, we think person names should not be generated, since they may not play an important role in an essay. Intuitively, in an essay about spring, "Nancy loves cherry blossoms" and "Sue loves cherry blossoms" are acceptable. Both sentences mean someone loves cherry blossoms, which are related to the theme "spring". Therefore, we extract person names using the language tool CaboCha, which can work as a named entity recognizer, then replace them with a special token [p] that means any person names. The model is trained to generate [p] as "[p] loves cherry blossoms." However, the accuracy of the named entity recognition by CaboCha is not high as we expected. In addition, we reconsider that person names are also important

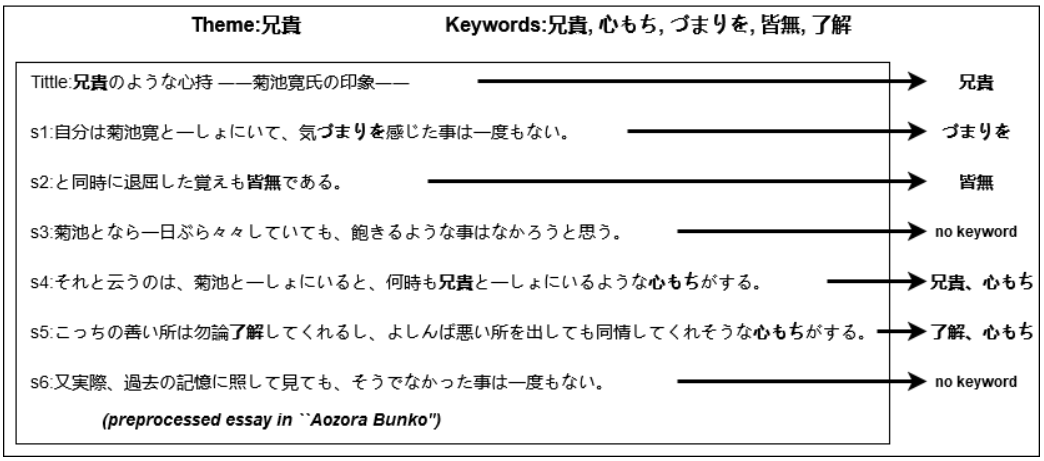

Figure 3.4: Example of extraction of theme and keywords

since they represent different individual characters in an essay. Therefore, we decide to handle person names as ordinary words.

Since one of the input in the CSGM and the output of the FSGM/CSGM are sentences, the retrieved essays are split into sentences by the punctuation mark "。". We also record the order of sentences in the paragraph in the essay. Then the sentences containing more than 78 tokens are removed. It means that the length of sentence in our EGM is limited to 80 including the start symbol $\langle sos \rangle$ and the end symbol $\langle eos \rangle$ to reduce the training time.

Extraction of Theme and Keywords

The last step of construction of the dataset is extracting the theme and keywords from the essays. We find that the title of the essay almost represents the main content of the essay. As a result, the theme is extracted from the title. Word segmentation is performed for the title of every essay with MeCab, and select the first noun other than a named entity as the theme of the essay. For example, as shown in Figure 3.4, 兄貴 (aniki; brother) is the first noun in the title of this essay, and it is not recognized as a named entity. We select it as the theme of this essay.

Next, we explain how to extract keywords for sentences. At first, several keywords are extracted from the essay. Rapid Automatic Keyword Extraction (RAKE) algorithm [17] and TF-IDF based method are tried. RAKE algorithm is a domain independent keyword extraction algorithm. It determines key phrases in a text by analyzing the frequency of word appearance and its cooccurance with other words in the text. In English, it extracts the key phrases of a document, but in Japanese, it is hard to split a document

or sentence into phrases. Therefore, we treat each word as a phrase. In the TF-IDF based method, we use all preprocessed essays to calculate IDF scores of the words. Then, we calculate TF scores of the words in each essay paragraph. Finally, words with the most highest TF-IDF scores are chosen as the keywords. Comparing the results of these two algorithms, TF-IDF based method achieved a better result. The top 5 keywords extracted by TF-IDF are set to the keywords of the essay. For example, 兄貴 (aniki; brother), 心もち (kokoromochi; feeling), づまりを (dumario[3]), 皆無 (kaimu; nothing), and 了解 (ryokai; alright) are five keywords extracted from the essay in Figure 3.4, which are shown in the top of this figure.

Contruction of Training Instances

After extraction of themes and keywords, training instances are built. Recall the FSGM and CSGM are the sequence-to-sequence models or the encoder-decoder models. Here, the input data of the encoder is called source data, and the output of the decoder is called target data. Using the extracted keywords, the sentences are selected to make instances of the training data. The keywords are searched in each sentence in the essay. If a sentence contains one of the keywords, firstly, it can be used as a training instance for the FSGM. That is, the sentence is chosen at the target data and the pair of the theme and the keyword is chosen as the source data. Table 3.3 shows the training instances extracted from the essay in Figure 3.4. As shown in the second row, the first sentence s_1 "自分は... " is chosen as the target data coupled with the theme "兄貴" and the keyword "づまりを" as the source data. In the target data, the special symbol $\langle sos \rangle$ representing the beginning of the sentence and $\langle eos \rangle$ representing the end of the sentence are added. In the source data, $[sep]$ is added as a separator between the theme and the keyword. Note that no training instance is made from a sentence that does not include any keywords. Two or more instances are made from a sentence including multiple keywords as shown in the 4-th and 5-th or 6-th and 7-th rows in Table 3.3.

Next, the sentence including the one of the extracted keywords is used as a training instance for the CSGM. That is, the sentence is chosen as the target data and the triplet of the theme, its previous sentence, and the keyword is chosen as the source data. Table 3.3 shows the training instances extracted from the essay in Figure 3.4. As shown in the second row, the second sentence s_2 "と同時に..." is chosen as the target data couple with the theme "兄貴", its previous sentence s_1 "自分は...", and the keyword "

[3]Actually, it is not a word. It is extracted by errors of word segmentation.

Table 3.3: Construction of dataset in the FSGM

source data	target data
兄貴 [*sep*] づまりを	⟨*sos*⟩ 自分 は...⟨*eos*⟩
兄貴 [*sep*] 皆無	⟨*sos*⟩ と 同時に...⟨*eos*⟩
兄貴 [*sep*] 兄貴	⟨*sos*⟩ それ と 云う...⟨*eos*⟩
兄貴 [*sep*] 心もち	⟨*sos*⟩ それ と 云う...⟨*eos*⟩
兄貴 [*sep*] 了解	⟨*sos*⟩ こっち...⟨*eos*⟩
兄貴 [*sep*] 心もち	⟨*sos*⟩ こっち...⟨*eos*⟩

Table 3.4: Construction of dataset in CSGM

source data	target data
兄貴 [*sep*] 自分 は...[*sep*] 皆無	⟨*sos*⟩ と 同時に...⟨*eos*⟩
兄貴 [*sep*] 菊池 と なら...[*sep*] 心もち	⟨*sos*⟩ それ と 云う...⟨*eos*⟩
兄貴 [*sep*] 菊池 と なら...[*sep*] 兄貴	⟨*sos*⟩ それ と 云う...⟨*eos*⟩
兄貴 [*sep*] それ と 云う...[*sep*] 了解	⟨*sos*⟩ こっち...⟨*eos*⟩
兄貴 [*sep*] それ と 云う...[*sep*] 心もち	⟨*sos*⟩ こっち...⟨*eos*⟩

心もち". Similarly, ⟨*sos*⟩ and ⟨*eos*⟩ are added as the beginning and end of the target data, and [*sep*] is added as a separator in the source data. Note that no training instance is made from the first sentence and the sentence without the keyword. Two or more instances can be made when the sentence includes multiple keywords.

The details of the constructed datasets for the FSGM and CSGM are shown in Table 3.5. More training instances are obtained for the FSGM. However, the file sizes are comparable since the length of the source data is short (always three) in the FSGM.

Construction of Test Data

The test data is also constructed to evaluate the EGM. Twenty essays are chosen, then theme and keywords are extracted in the same way in the construction of the training data. Each instance in the test data consists of

Table 3.5: Statistics of datasets

	Dataset for FSGM	Dataset for CSGM
# of data	294,428	170,316
Average # of tokens (source)	3	35
Average # of tokens (target)	29	34
File size	61.5 MB	60MB

one theme and four keywords. The EGM is applied to the test data and the generated essay (four sentences) is manually evaluated. Note that these 20 essays are not used to construct the training data. Furthermore, any instances are removed from the training data if its source theme and keywords are in the test data. It ensures that the training and test data are completely mutual exclusive.

3.2.3 Training

This subsection explains details of training of the FSGM and CSGM. All models in our experiment are implemented by Python3 in Jupyter notebook. We use PyTorch, one of the Python machine learning libraries, to realize our all models. We train the models on a single RTX 5000 GPU.

Our model has the 512-dimensional hidden states and 512-dimensional word embedding vectors. The sizes of the vocabulary used for the FSGM and CSGM are 72,758 and 69,698 words, respectively. The vocabulary is shared in both the source data and target data. The vocabulary is made by a set of all words appearing in the training data. In other words, the word whose frequency is greater than or equal to one in the training data is added into the vocabulary.

Since this is the first attempt in the Japanese essay generation task, we set relatively large parameters (512 dimensions). In addition, the additional parameters are required for the attention mechanism and the coverage mechanism. Therefore, the numbers of the parameters in our models are more than those in a normal sequence-to-sequence model: 118,583,808 parameters for the FSGM and 113,883,648 parameters for the CSGM.

Unlike other NLG models, pre-trained word embedding vectors are not used in all models. They are learned through the training of the sequence-to-sequence model. We use AdamW (Adam with decoupled weight decay) [10] as the optimization algorithms. A lot of learning rates were tried, and finally, we found that the learning rate 8.0×10^{-4} for the FSGM, and 5.0×10^{-4} for the CSGM could yield relatively better results. In all models, gradient clipping is set with a maximum gradient norm of 1. A learning rate scheduler is applied to adjust the learning rate size during training. The learning rate is reduced by a factor of 2 once when a metric has stopped improving. The cooldown time is set to 3. Another optimization algorithm SGD (Stochastic Gradient Descent) [21] was also tried, but the generated sentences were very poor.

As described in Subsection 3.2.2, during training, the length of the sentences in the source and target data is limited to 80 tokens (including the start token $\langle sos \rangle$ and the end token $\langle eos \rangle$). Due to the limitation of the

number of the token in the training, in the test, the maximum length of the generated sentence is limited to 79 (except the start token $\langle sos \rangle$). The decoder stops generating the sentence when the number of the output tokens reaches 79 or the special token $\langle eos \rangle$ is generated. The beam search is often used for a sequence-to-sequence model since it is often effective to improve an output sequence. However, in our preliminary experiment, the use of the beam search makes the model generate sentences that are short as well as incoherent with the theme, keyword, and other sentences. Finally, we decide not to use the beam search.

The batch size is set to 88 for the FSGM and 80 for the CSGM. The FSGM is trained by 40 epochs, while the CSGM by 50 epochs. Since number of the training data for the FSGM is greater than the CSGM as described Subsection 3.2.3, the fewer number of the epochs is set for the FSGM. Initially, the number of the epochs was set to 100, but it requires too long time. So the number of the epochs was decreased. The training of the FSGM took 2 days and 12 hours, while the training of the CSGM took 2 days and 1 hour. Since the number of the parameter in the CSGM is a little smaller than the FSGM, the training time of the CSGM is a little faster. In this experiment, as a result, the teacher rate is set as 1 in training and 0 at test.

In general, the hyperparameters (such as the number of epochs, the learning rate, the dimension of the hidden state, and so on) are often optimized on development data in deep learning. However, in the task of the essay generation, the automatic evaluation of the essay is hard and almost impossible. Therefore, the hyperparameters are determined in an ad-hoc manner in this study. It is one of the important future work.

3.3 Theme-Attention Essay Generation Model

3.3.1 Framework

In this section, we will introduce our modified model Theme-Attention Essay Generation Model (TA-EGM). This model is based on the EGM, but the theme is separately inputted as the weight of attention. Figure 3.5 shows the sequence-to-sequence model in the TA-EGM. The most important difference with the EGM of Figure 3.2 is that the theme is given as the attention in the encoder, not in the input sequence. Similar to the EGM, this sequence-to-sequence model is used as the FSGM and CSGM. Hereafter, we call First Sentence Generation Model and Content Sentence Generation Model in TA-EGM as FSGM-TA and CSGM-TA, respectively.

This model is inspired by the Hierarchical Attention Networks (HAN)

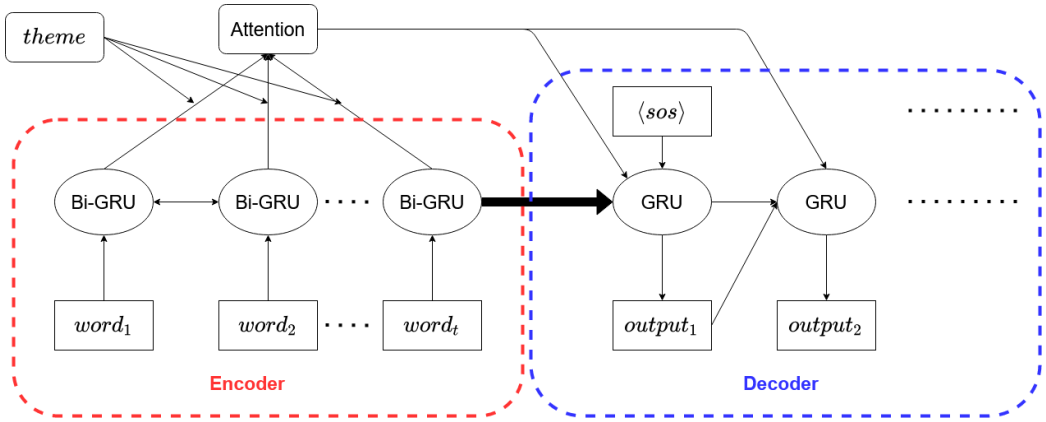

Figure 3.5: Sequence-to-sequence model in TA-EGM

presented by Zhang et al. [28]. They encoded extracted aspect words as continuous real-valued vectors, then added them into vectors of input words to make the input information richer and to improve the ability to capture related words. In our model, we also hope the model generates words that are related to the theme so that generated sentences can be coherent on the theme. That is the reason why the mechanism of the HAN is applied to our model.

The following equations show modified parameters in the TA-EGM.

$$\hat{h_j} = W_g^\top [u; h_j] \tag{3.14}$$

As shown in Equation (3.14), to enhance the relevance between the theme and generated sentences, the embedding of theme u is added as a part of the output of the encoder h_j to obtain $\hat{h_j}$. Then, $\hat{h_j}$ will be used to replace the h_j from the Equation (3.5) to (3.7) to calculate in the attention mechanism.

The input sequence is modified as follows. First, the input of the FSGM-TA is defined as Equation (3.15).

$$[\ k_1\ ], \quad th \tag{3.15}$$

It means that only the keyword k_i is given as the input sequence, while the theme th is given as the attention. Second, the input of the CSGM-TA is defined as Equation (3.16).

$$[w_1^p \cdots w_n^p \quad [sep] \quad k_i], \quad th \tag{3.16}$$

It means that a previous sentence and a keyword is given as the input sequence, while the theme is separately given as the attention.

3.3.2 Training

In the TA-EGM, there are 512-dimensional hidden states and 512-dimensional word embeddings. The training data and the vocabulary are the same as the EGM. Since the additional attention weight are used, the number of the parameters is more than the EGM: 156,883,969 in the FSGM-TA and 150,625,281 in the CSGM-TA.

Word embeddings are also learned during training of the overall model. AdamW is also used as the optimization algorithm. The learning rate is set to 8.0×10^{-4} in the FSGM-TA, and 6.0×10^{-4} in the CSGM-TA. The same gradient clipping and learning rate scheduler are set as the EGM. Since the same training data as the EGM is used, the length of the sentences in the source data and target data is limited to 80 tokens (including $\langle sos \rangle$ and $\langle eos \rangle$). In the test, the maximum length of the generated sentence is limited to 79 (except for $\langle sos \rangle$) with no beam search. The batch size is set to 64 for the FSGM-TA and 86 for the CSGM-TA. The FSGM-TA is trained by 40 epochs, while the CSGM-TA is trained by 50 epochs. The training of the FSGM-TA took 2 days and 14 hours. while the CSGM-TA took 2 days and 12 hours. The teacher rate is set to 1 in training and 0 at test.

3.4 Baseline

Our proposed model is compared with a baseline model. We define the baseline is a model that generates all sentences from a given theme and keywords. Independently generated sentences are concatenated to form an output essay. The structure of the baseline model is the same as the FSGM (Figure 3.2) in the EGM. Referring to other text generation experiments, in the input sequence, the symbol $[sep]$ between the theme and the keyword is not used as shown in Equation (3.17).

$$[th \quad k_1] \tag{3.17}$$

For the baseline model, another training data is created. It is created in the same way to construct the training data for the FSGM. The size of vocabulary of baseline is 72,740, which is the number of words in the dataset. The number of the training instance is 293,893. It is slightly smaller than the number of the training data for the FSGM shown in Table 3.5. Recall that we remove training instances if the theme and keywords appear in the test data. By this procedure, the number of the training data is different.

The number of the parameters in the baseline model is 118,556,160. Pre-trained word embedding is not used, either. AdamW is also used in the

baseline model. The learning rate is set to 4.0×10^{-4}. We use the same gradient clipping and learning rate scheduler used for the FSGM in the EGM. Also, during training, the sentence length in the source data and target data is limited to 80 tokens (including $\langle sos \rangle$ and $\langle eos \rangle$). In the test, the maximum length of the generated sentence is limited to 79 except for $\langle sos \rangle$ with no beam search. The batch size is 86. The teacher rate is set to 1 in training and 0 at test. The number of the epochs is set to 40. The training took 2 days and 13 hours.

Chapter 4

Evaluation

This chapter presents evaluation of the proposed method of essay generation. In Section 4.1, we introduce the details of evaluation methods. In Section 4.2, results of the evaluation of essays generated by the proposed models are introduced and discussed. Finally, in Section 4.3, we discuss the limitations of the models from the results of the evaluation.

4.1 Experimental Setting

In the past studies of text generation such as automatic summarization and storytelling, automatic evaluation has been often carried out. In automatic evaluation, ground truth data is prepared by human. For example, in summarization, a summary written by human subject is prepared as the ground truth for each original document. Then the similarity between the ground truth and texts generated by a system is measured by several criteria such as BLEU and ROUGE. However, in the essay generation, the ground truth is rather hard to prepare, since an essay is rather free text related to a given topic. In other words, there exists many acceptable essays for a topic. It is almost impossible to enumerate such acceptable (or good) essays in advance.

Therefore, in this experiment, automatically generated essays are evaluated by human subjects. They are asked to answer questionnaire to evaluate the quality of essays from several points of view. Questions are made following the Likert Scale, i.e. a five point scale. The subjects are required to give a score between 1 to 5 for a question. Four aspects are considered: the first three evaluate individual sentences in an essay, while the last evaluates the overall essay.

The first aspect is the comprehensibility of the sentence. Simply to say, a human subject evaluates whether the generated sentence is natural. In this

Table 4.1: Question A: sentence comprehensibility

Q	Comprehensibility of sentence (Is the sentence natural?)
1	The sentence is not grammatically correct and can not be understood it at all.
2	The sentence is generally grammatically correct, but the meaning can not be understood very well.
3	Can not say either.
4	The sentence is generally grammatically correct, but the meaning is understood somehow.
5	The sentence is grammatically correct and can be understood.

Table 4.2: Question B: relevance to theme

Q	The relevance between the sentence and the theme (Is the sentence related to the theme?)
1	Not related at all.
2	Not related well.
3	Can not say either.
4	A little related.
5	Related well.

Table 4.3: Question C: relevance to keyword

A	The relevance between the sentence and the keyword (Is the sentence related to the keyword?)
1	Not related at all.
2	Not related well.
3	Can not say either.
4	A little related.
5	Related well.

Table 4.4: Question D: coherence in essay

Q	The coherence in the whole generated essay (Are the sentences in the essay coherent?)
1	Not at all.
2	Not very consistent.
3	Can not say either.
4	A little consistent.
5	Almost completely consistent.

aspect, 2 items are evaluated: one is whether the grammar of the generated sentence is correct, the other is whether the generated sentence is meaningful. These two characteristics are directly proportional. If there are many grammatical errors in a sentence, it is difficult to understand its meaning. Table 4.1 shows the question for the comprehensibility. Hereafter, we refer it as Question A.

The second aspect is the relevance between the generated sentence and the given theme. Recall that in our system an essay is generated for a given theme and keywords. This aspect evaluates whether the generated sentence properly describes something about the topic given by the user. It is hard to evaluate automatically, but human can understand the meaning of the sentence. Therefore, it is possible to evaluate whether the theme appears in the generated sentence or not. Table 4.2 shows the question and five point scale for relevance to the theme. Hereafter, we refer it as Question B.

The third aspect is similar to Question B. The relevance between the generated sentence and the keyword is evaluated. Again, recall that each sentence is generated from a given keyword. This aspect evaluates whether the generated sentence properly describes some contents about the keyword given by the user. Table 4.3 shows the question and five point scale for relevance to the keyword. Hereafter, we refer it as Question C.

The last aspect is coherence in the whole generated essay. We consider this is the most significant item in this evaluation, since one of our goals in this study is to improve the coherence in the generated essay. In this question, the subject needs to answer whether the whole essay is consistent or individual sentences are not related each other. It is different from the other questions that should be answered for every generated sentence. It is asked for the overall essay. The subject should read all sentences, then given a point from 1 to 5 to evaluate the coherence. Table 4.4 shows the question and five point scale for coherence in the essay. It is called Question D hereafter.

The following three models are compared in the experiment.

- Baseline model (BL)
  The system described in Section 3.4. It produces an essay by generating four sentences independently, where a theme and keyword are given as input. Unlike our two proposed models described below, all sentences are generated by the same model.

- Essay Generation Model (EGM)
  The system described in Section 3.2. It produces an essay by generating four sentences one by one. The first sentence is generated by the FSGM,

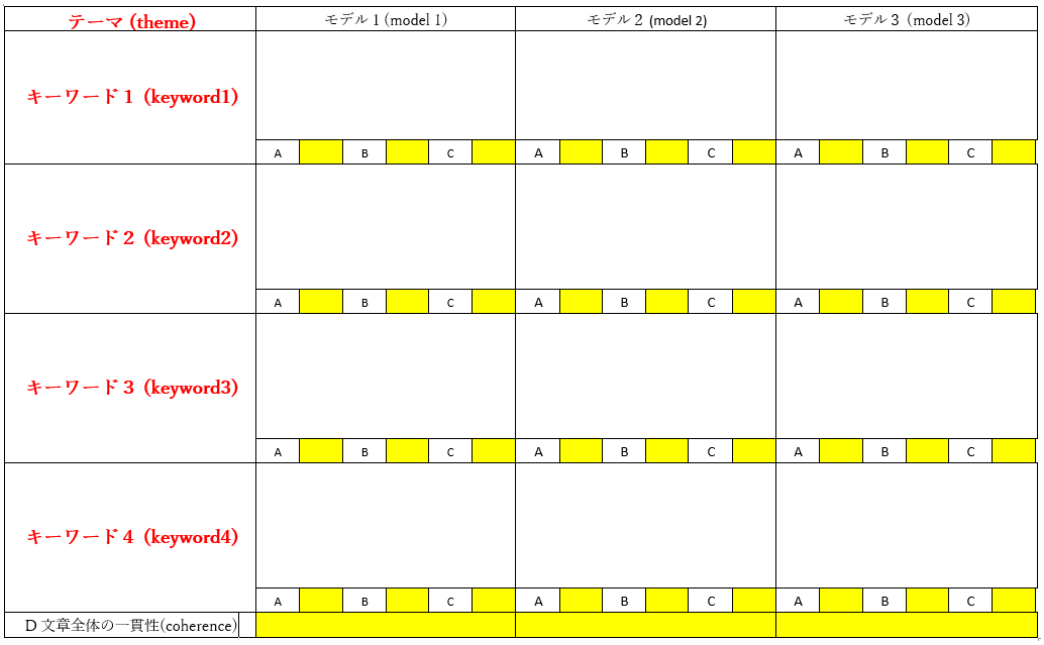

Figure 4.1: Evaluation table

while the rest by the CSGM. In the generation of the sentence, a theme and keyword as well as a previously generated sentence (except for the generation of the first sentence) are given as the input.

- Theme-Attention Essay Generation Model (TA-EGM)
  The system described in Section 3.3. Similarily, it produces an essay by generating four sentences. The first sentence is generated by the FSGM-TA, while the rest by the CSGM-TA. In the generation of the sentence, a keyword and a previously generated sentence is given to the sequence-to-sequence model, while a theme is given as an attention.

The questionnaire is made up so that these essay generation models are in blind. The subjects do not know which model is used to generate an essay. Furthermore, the order of models for which the subjects answer the questions is randomly changed for each test instance. It can ensure the fairness and authenticity of the evaluation.

Figure 4.1 shows the evaluation sheet used for the questionnaire. The blanks with a yellow background are filled with the scores by human subjects. The theme and keywords are in the first columns, while the identifier of the models are shown in the first row where the actual models are hidden. Scores of Question A, B, and C are filled after every generated sentence, while the score of Question D is filled in the last of the table.

Four human subjects participate in the evaluation. All subjects are native Japanese speakers. As already explained, the number of the essay in the test data is 20. For each essay, each of three models generates 4 sentences. Therefore, the total number of the sentences to be evaluated is $20 \times 4 \times 3 = 240$. In addition to these sentences, the subjects are asked to answer Question D for $20 \times 3 = 60$ essays.

Using this evaluation table, we evaluate the whole generated essays and sentences at the same time. Our questionnaire is designed to clearly and comprehensively understand the quality and shortcomings of our models. Investigation of results of the questionnaire could enable us to solve existing problems and improve performance resulting in development of a better essay generation model.

4.2 Results

Figure 4.2 shows the examples of the essay generated by three models, the baseline, EGM, and TA-EGM. In this example, "病"(*yamai*;disease) is given as the theme, and "桜"(*sakura*;cherry blossoms), "春"(*haru*;spring), "待ち"(*mati*;wait), and "単調"(*tantyo*;dullness) are given as the keywords. The essays are generated as the order of s_1, s_2, s_3 and s_4. Each sentence is generated from the keyword written in the brackets in order. The 4 sentences are finally combined into an essay of each model.

From Figure 4.2, we can see that most of sentences in three models are with a little grammatical error. Although the relevance between the sentence and the keyword or the theme is not bad, it is hard to say that they are excellent. The repetition is still found in several sentences.

Table 4.5 shows the results of the evaluation of the essay generation models. The first table shows the average scores of 4 subjects for the Question A, B, and C, which evaluate the quality of each sentence. The columns s_1, s_2, s_3 and s_4 indicate the scores for the first, second, third and fourth generated sentences respectively, while "ave." indicates the average of these four sentences. The second table shows the average score of the subjects for the Question D that evaluates the consistency of the overall essay.

First, we discuss the results of the consistency of the generated essays(i.e. the Question D). Figure 4.3 is the scores of the coherence among the whole generated essay. It is the same as the results of the Question D in Table 4.5. We can see that our EGM is slightly better than the baseline by 0.12 point. This means that our EGM can generate better coherent essays. In our model, the idea that makes information flow from the previous sentence to the following sentence works well to some extent. Furthermore, comparing to

models	Theme: 病 Keywords:桜、春、待ち、単調	
baseline model	s_1 (桜)	桜の花の頃萎れたあたりは、桜の咲いた桜の幹に枝を置いてはいないらしく、三寸から三十近いせいぜい三発ぐらいの桜を、植木屋のなかからは冷たい黒い鳥が這入っている。
	s_2 (春)	春から夏の日には、庭の輪の上に病が多いのかと思って感じた。
	s_3 (待ち)	私の頭の中には夜の寒さの瞬間の日の会話がかなりの気分になって、その気分はどうかと待ち伝えながら居ました。
	s_4 (単調)	それから内へ出てみると、内に前の親類の談話会や、森の小路にも来て、底が深いのを感じました。
Essay Generation Model	s_1 (桜)	桜には桜だけれど、私に小さな桜があっても春の花が白く散って見えた頃は、桜の花の露が枝を押して来て、病中に共通な桜の花を思い比べた。
	s_2 (春)	桜は春よりも丘の上で桜若葉を盛り、あたりを桜町の祭にする小舟で、今では桜の花を見たことはない。
	s_3 (待ち)	遠い昔、桜の花は最初に中腹にある。
	s_4 (単調)	東には風が吹くようであるが、それでも汽車はまだ吹き過ぎた。
Theme-Attention Essay Generation Model	s_1 (桜)	だが、桜の花の開きは確かに外にあるのに、急に春の近づく時と同様にひどく混雑するのだから、すぐ私も色の悪かったのは、非常な桜の花よさに過ない停車場である。
	s_2 (春)	だから桜の方は、安全で実のどこにもないので、桜の樹の横から桜のように長く空に浮んで来る。
	s_3 (待ち)	桜は春のこんもりした桜の実を微して多くの桜堤に続いて、桜がたの桜一つに多くある。
	s_4 (単調)	桜の枝よりも伸び上っているあの桜の枝の中にも、桜の花の美しい桜が咲いている外はない。

Figure 4.2: Example of generated essays

Table 4.5: Results of experiment

Question	Model	s1	s2	s3	s4	ave.
A(naturalness)	Baseline	2.84	3.10	2.59	2.51	2.76
	EGM	2.93	2.96	2.83	3.23	2.98
	TA-EGM	3.09	2.80	2.74	2.83	2.86
B(theme)	Baseline	2.69	2.31	1.81	1.89	2.18
	EGM	2.24	1.93	1.60	1.86	1.91
	TA-EGM	2.83	1.78	1.70	2.00	2.08
C(keyword)	Baseline	2.90	2.30	1.93	2.19	2.33
	EGM	2.66	1.96	1.71	1.80	2.03
	TA-EGM	3.05	1.73	1.70	1.93	2.10

Question	Model	essay
D(coherence)	Baseline	1.94
	EGM	2.06
	TA-EGM	2.25

the baseline model, the improvement by the TA-EGM is 0.31 point. It also outperforms the EGM by 0.19 point. It can be seen that giving the theme by the attention mechanism makes the model capture more information about the topic. It indicates that the attention mechanism is an effective way to reflect the theme in the contents of the generated sentences, resulting in the improvement of the coherence in the overall essay.

Next, we discuss the quality of the indivial generated sentences. Figure 4.4 is the average of scores for Question A (comprehensibility) in all three models. It is the same as the average score for the Question A in Table 4.5. Our EGM achieves the best. However, we consider that the generated sentences of these three models are similar with respect to grammatical correctness.

Figure 4.5 shows the average scores for the question about the relevance to the theme. It is the same as the average scores of the Question B in Table 4.5. In the EGM, the relevance between the generated sentences and the theme is lower than the baseline model. It may be caused by the difference of the inputs of the models. In the EGM, when generating the sentences after the first sentence, the previously generated sentence is added as the input. It makes the model generate a sentence that is more related to the previous sentence rather than the theme. At the same time, it is unable to guarantee the quality of the previous sentence used as the input of the CSGM. In other words, since the previous sentence is automatically generated, it is not good

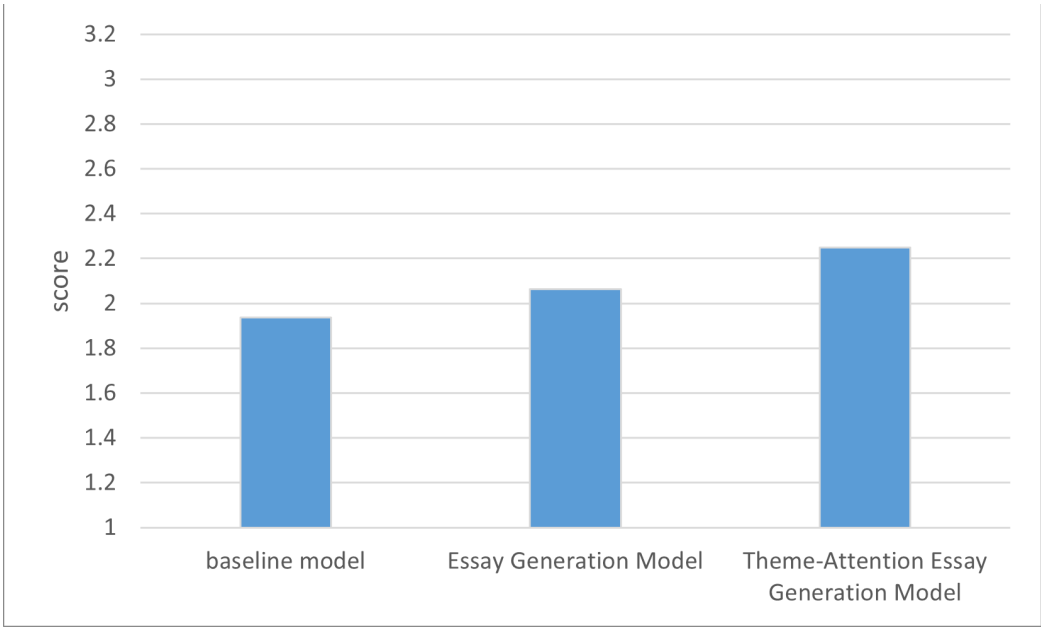

Figure 4.3: Scores of coherence in three models

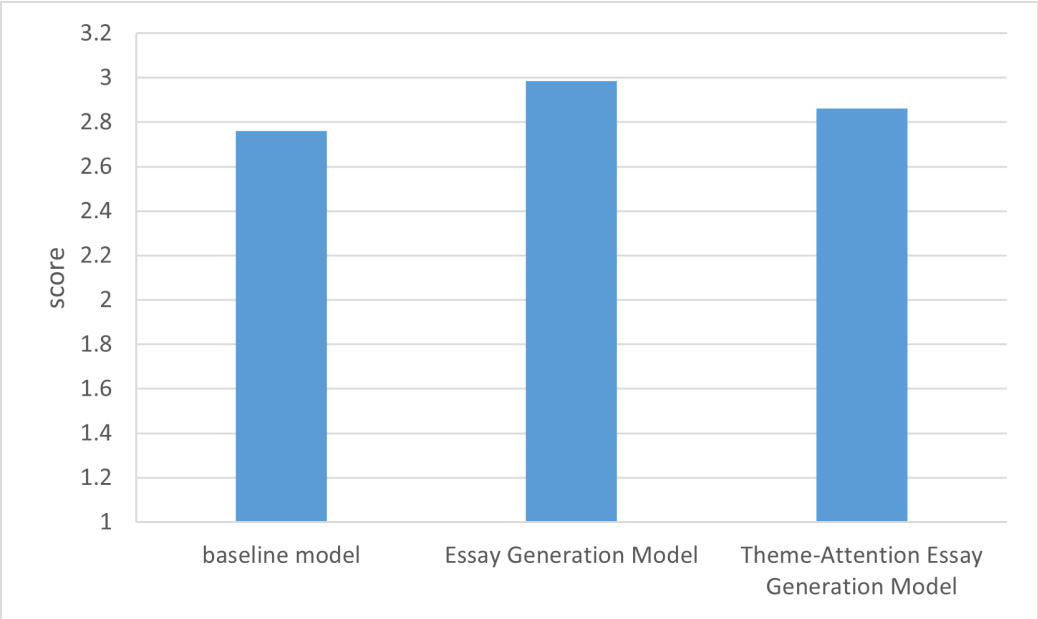

Figure 4.4: Scores of comprehensibility in three models

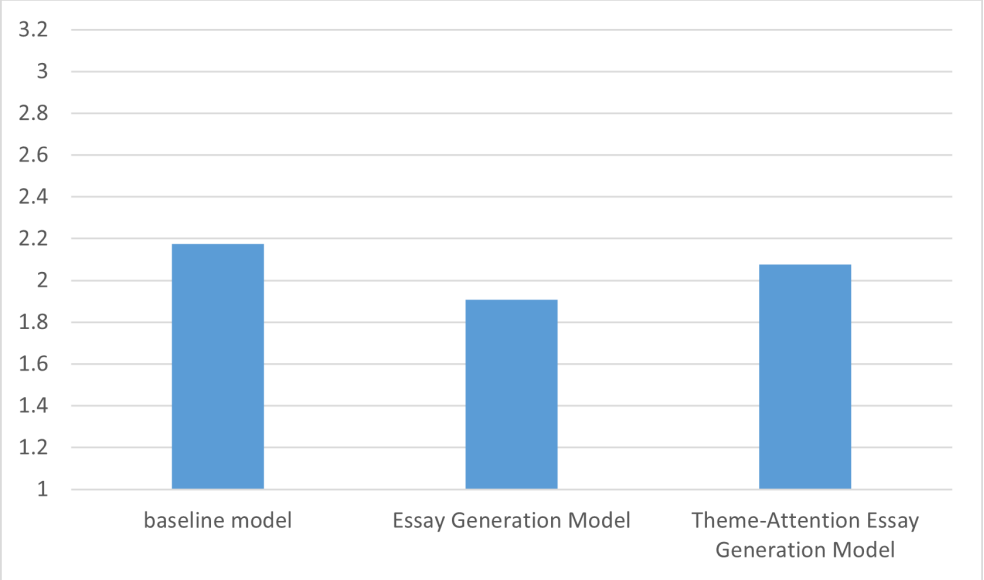

Figure 4.5: Scores of relevance to theme in three models

as one written by human. On the other hand, in the baseline model, since only the theme and keyword (not a noisy previous sentence) are given, it tends to generate sentences that are relevant to the theme. We guess that the above facts are the major reasons of the reduction of relevance between the generated sentences and the theme. However, when the theme is given via the attention mechanism in the TA-EGM, the score of relevance between the generated sentence and the theme is almost the same as the baseline model. The difference of these two models are only 0.1 point. Therefore, the TA-EGM can contribute to improve the relevance between the generated sentences and the theme.

Figure 4.6 shows the average scores for the question about the relevance to the keyword. It is the same as the average scores of the Question C in Table 4.5. The scores of the EGM and TA-EGM are almost the same. It may be because the way how to give the keyword in the sentence generation is the same in both models, i.e. the keyword is entered as the input of the sequence-to-sequence models. However, the scores in these proposed models are worse than the baseline. Similar to the scores of the relevance to the theme shown in Figure 4.5, the relevance to the keyword is lost by giving a low-quality previous sentence as the input in the EGM and TA-EGM.

Next, the quality of the sentences in the first, second, third and fourth sentence is evaluated individually. Recall that the different models, FSGM or CSGM in the EGM and FSGM-TA or CSGM-TA in the TA-EGM, are used

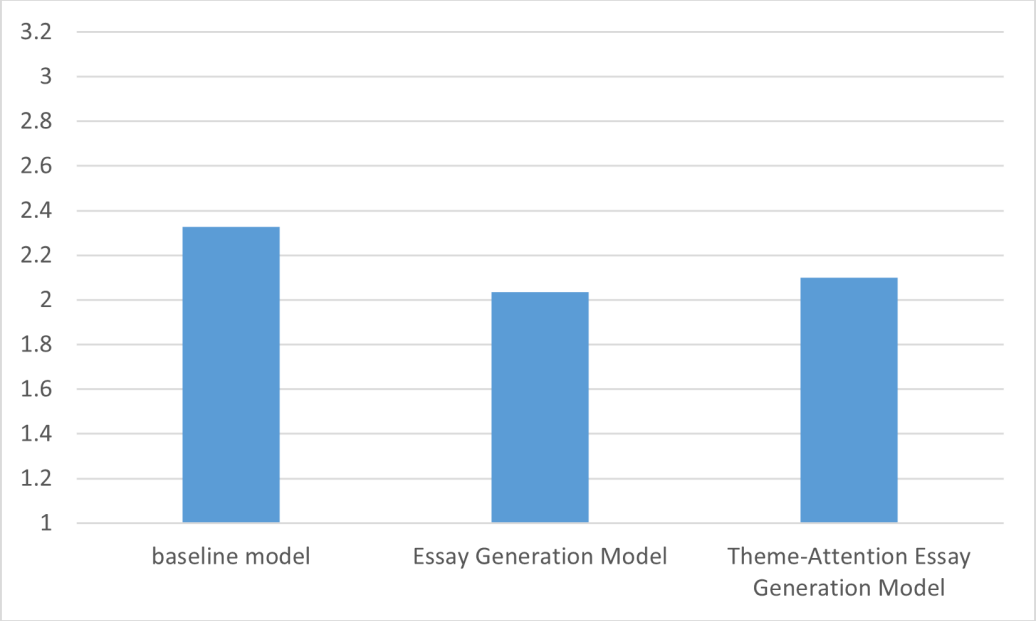

Figure 4.6: Scores of relevance to keyword in three models

in the generation of the first sentence or the rest. Therefore, the performance of the generation of the first sentence and the rest of the sentences should be compared. Figure 4.7, 4.8 and 4.9 are the graphs showing the average scores of the first to fourth sentences (indicated by s_1 to s_4) with respect to the comprehensibility (Question A), relevance to the theme (Question B) and relevance to the keyword (Question C), respectively.

Seeing Figure 4.8 and 4.9, we find clear difference of the scores of both relevance to the theme and keyword. That is, the scores of the first sentence are greater than that of the rest of the sentences. As for the EGM and TA-EGM, we consider it is caused by the features of our FSGM (or FSGM-TA) and CSGM (or CSGM-TA). In the FSGM and the FSGM-TA, the first sentence is generated only by the theme and the keyword without the previously generated sentence. It ensures the attention during generation on the theme and the keyword. However, the input is different in the CSGM and CSGM-TA, in which the previously generated sentence is used as the input of the models. Since the models consider not only the theme and the keyword, they tend to generate sentences that are less relevant to the theme and keyword. Another problem in the CSGM and CSGM-TA is that the use of the (bad) previous sentence increases the negative influence on the sentence generation at the later stage. If the quality of the sentence entered to the model is low, the quality of the current sentence becomes worse than

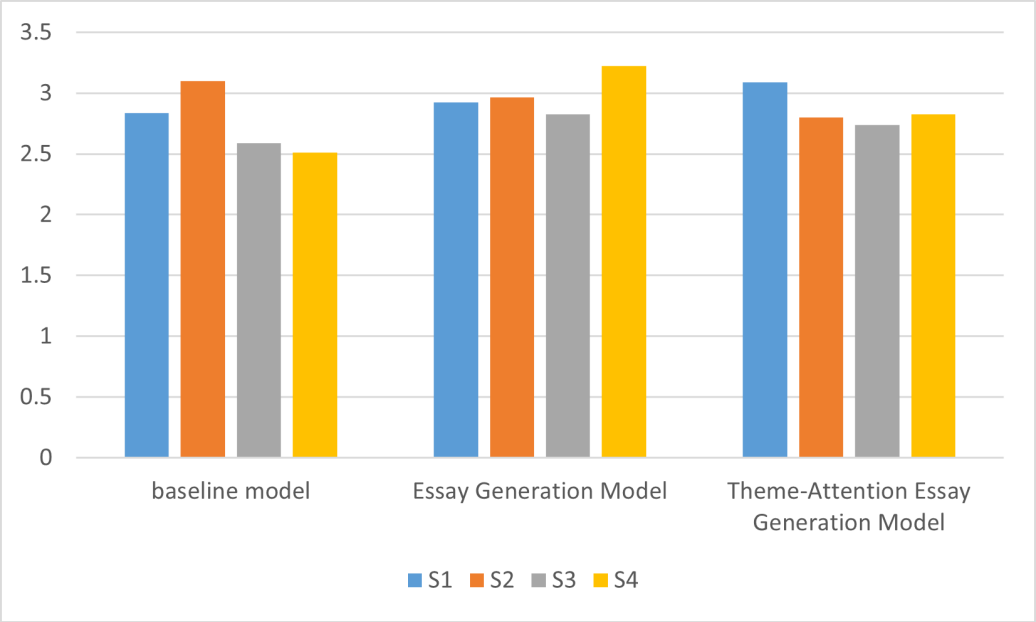

Figure 4.7: Comparison of scores of comprehensibility individual sentences

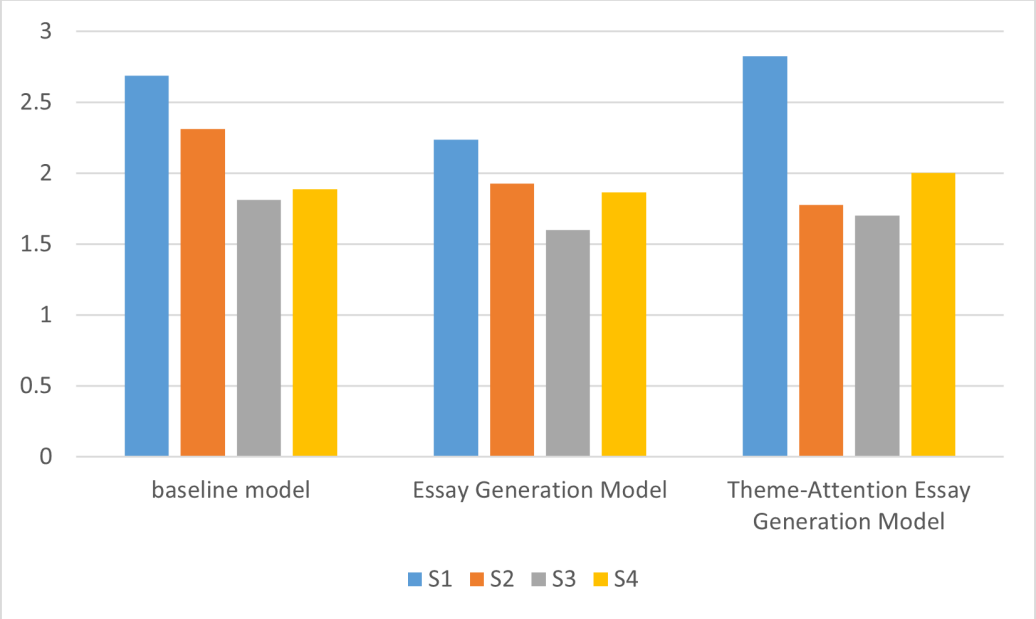

Figure 4.8: Comparison of scores of relevance to theme individual sentences

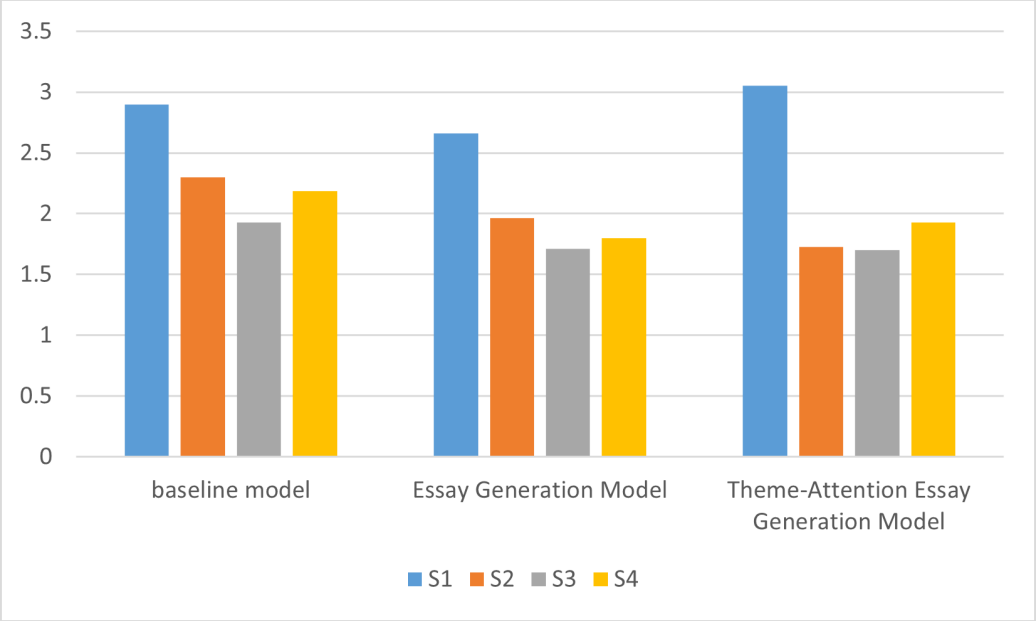

Figure 4.9: Comparison of scores of relevance to keyword individual sentences

the previous sentence. Once the low-quality sentence is generated by the model, its negative impact is propagated into the generation of succeeding sentences, i.e. the third and fourth sentences. Although we cannot confirm a clear tendency that the scores becomes worse in the later sentences in Figure 4.8 and 4.9, we guess it is one of the reasons why the scores of the relevance to the theme and the keyword for the second, third and fourth sentences are lower than the first sentence.

Unlike our proposed models, all sentences are equally generated by the baseline. The previous sentence is always not given as the input; the theme and keyword are given for all sentences. However, it is found that the score of the first sentence is better than the rest of the sentences in the baseline model. Currently, we cannot find any clear reason why.

Finally, we evaluate variance of four subjects in the evaluation of the essays. Table 4.6 shows the average scores for the questions given by each subject in the columns of "Subj1", "Subj2", "Subj3" and "Subj4", and the standard deviation of four subjects in the column of "SD". Since the Question A is possible to be evaluated objectively, the standard deviation for Question A is lower than other 3 questions. Since the evaluation of Question B, Question C, and Question D is vulnerable to the subjective factors of the subjects, the standard deviations are not as low as Question A. Overall, the standard deviations of our models are lower than the baseline model. It indicates that

Table 4.6: SD of scores of questions in three models

Question	Model	Subj1	Subj2	Subj3	Subj4	SD
A(naturalness)	BL	2.51	3.34	2.54	2.65	0.3378
	EGM	2.71	3.58	2.80	2.85	0.3445
	TA-EGM	2.63	3.35	2.71	2.76	0.2857
B(theme)	BL	2.36	2.58	1.46	2.30	0.4238
	EGM	1.63	2.43	1.49	2.0875	0.3730
	TA-EGM	1.99	2.61	1.55	2.15	0.3801
C(keyword)	BL	1.40	3.19	2.25	2.48	0.6379
	EGM	1.50	2.69	1.81	2.14	0.4393
	TA-EGM	1.46	2.80	1.89	2.25	0.4909
D(coherence)	BL	1.30	2.40	1.65	2.40	0.4788
	EGM	1.50	2.35	1.95	2.45	0.3748
	TA-EGM	1.70	2.65	2.25	2.40	0.3482

our model can generate essays of stable quality.

4.3 Discussion

One of the most important results shown in the previous section is that our proposed models can achieve the better performance on generating a coherent essay than the baseline. In other words, our methods can produce coherent sentences that are related to each other. Especially, our TA-EGM can get the highest score of the coherence in the essay among three models. At the same time, its score of the relevance to the theme and keywords are almost equivalent to the baseline model. Nevertheless, we found our proposed models had several shortcomings during our experiment. In this section, we will discuss these disadvantages.

The first one is the decline of the relevance to the theme and keyword in the sentence generation after the first sentence. As already discussed, in our model, this is inevitable since we feed the previously generated sentence, which are often not good, into the encoder in the CSGM and CSGM-TA. To solve this problem, it is necessary to find another way to pass the information of the previous sentence into the next encoder in order to keep the coherence among sentences, instead of giving the whole previously generated sentence.

The second disadvantage is the theme and keywords used for training the models are not very appropriate. In our datasets, the theme is extracted from the title of the essay, and keywords are extracted from sentences. They are all nouns. However, the automatically extracted theme sometimes does not

represent the central content of a whole essay. Although a noun that firstly appears in the essay title is extracted as a theme, a phrase may be often better than just a noun. We should investigate a better way how to appropriately extract a noun or a noun phrase as a theme. One of the possible directions is to apply machine learning as well as to extract a theme from not only a title but also a body text of an essay. As for extraction of the keywords of sentences, a type of a sentence should be considered. We observed that the sentences in essays were roughly divided into two categories, "description" and "action". A sentence of "description" is used to describe a thing, such as the sentence "The maple leaves are so beautiful." In this sentence, "maple" can be an appropriate keyword. On the other hand, a sentence of "action" represents an action performed by a human, such as the sentence "I washed the dishes today." In this sentence, the word "dish" can not be suitable as a keyword. The key phrase "wash the dishes" might be better as the representation of the key concept of the sentence. In future, how to extract keywords or key phrases from sentences must be explored to improve the relevance between the generated sentence and the keyword. In addition, we expect that improvement of the relevance to the keyword can also improve the comprehensibility of the generated sentences.

The third disadvange is the global coherence is not enough. Although both proposed models can improve the coherence in generated essays, it is still at a relatively low level. The average scores are around 2 in 5-point scale. Therefore, we have to find better methods. In our model, the theme and the previously generated sentence are only used to control the coherence. Obviously, it is not enough. Many strange, meaningless or unreliable sentences are found in the essay generated by our models. Another way to give constraints on the contents in the sentence generation is required to produce consecutive coherent sentences. One of the possible constraints can be time, location, or other things. Such constraints can be given as the input of the sequence-to-sequence model instead of giving the whole previous sentence. With these constraints, the relevance among the sentences in the generated essay becomes natural and faithful, as an essay written by a human.

Chapter 5

Conclusion

5.1 Summary

This thesis focused on a new task of automatic generation of an essay in Japanese, and proposed a novel method based on deep learning for it. The contribution of the thesis could be summarized as in the following three features.

Firstly, we proposed the Essay Generation Model (EGM) based on the sequence-to-sequence model with the attention mechanism and coverage mechanism. Using the theme and keywords given by the user, the model automatically generated an essay by producing sentences one by one. The model consisted of two sub-models: the First Sentence Generation Model (FSGM) and the Content Sentence Generation Model (CSGM). In the FSGM, the first sentence in an essay was generated from the theme and the first keyword given as the input data. In the CSGM, the rest of sentences in an essay were generated from the theme, the keyword and the previous sentence generated by the model. Comparing to the baseline model, this model generated better coherent essays where individual sentences were not totally independent but related each other as a story.

Secondly, the Theme-Attention Essay Generation Model (TA-EGM) was proposed to improve the coherence in overall essay. The EGM was modified so that the theme was not given as the input of the encoder but was passed to the model via the attention mechanism. The theme was transferred into the vector at each timestep in the encoder to enhance the relevance of the theme and words generated in the decoder. Similar to the EGM, it consists of two sub-models: the FSGM-TA used for generation of the first sentence and the CSGM-TA used for the rest of sentences. With this model, the coherent in the overall generated essay was improved.

Finally, we created the new corpus for Japanese text generation. The 2,140 documents written in the new writing style were downloaded from the website "Aozora Bunko" under the category of "Essay, Review". Then various preprocessing was performed to clean up the texts, such as conversion from old Japanese characters to new ones. For each essay, the theme was extracted from the title of the essay. Then, five keywords were extracted from the essay based on the TF-IDF based scoring. In this corpus, each data contained a preprocessed essay, a theme, and 5 keywords. Two training datasets were derived from the corpus: one was used to training the FSGM and FSGM-TA, the other was for the CSGM and CSGM-TA.

5.2 Future Work

Although this thesis has got some achievement in the automatic generation of essays, especially improvement of the consistency, the quality of the essays was far from ones written by human. There is much room to revise and improve our models. The major problems to be solved in future can be summarized as the following three aspects:

Improvement of Dataset for Essay Generation

The datasets used to train our model should be revised. First, the theme should be extracted more accurately. As we discussed in Section 4.3, in our current training data, the themes of the essays were often inappropriate. A wrong word was selected from the title as the theme, or a chosen noun did not represent the center of the essay. It may cause declines of the quality of the generated sentences as well as the relevance between the theme and the generated sentence. The readability and consistency of the generated essays become also poor. It would be better to select words from not only the title but also the sentences in the essay as the theme of the essay. Next, it is required to improve how to extract the keyword that represents the content of the sentence. It is similar to the problem on the extraction of the theme, but is more difficult. Because it is uncertain what kinds of lexical units (a word or a phrase) are appropriate as the representation of the core meaning of the sentence. As discussed in Section 4.3, either a keyword or key phrase can be more appropriate depending on the type of the sentence. Third, more modern essays should be collected to create the dataset. The essays used in the current dataset were a little old. Some writing styles were not used at all now. It was hard to transfer them into the new writing style with no error Obviously, the quality of the corpus or the training data heavily influences

the performance in the essay generation

Reconsideration of Input Data

As we explained in Section 3.1, in the CSGM and CSGM-TA, the previously generated sentence was added as the input of the encoder in the sequence-to-sequence model. Although it could contribute to enhance the coherence in an essay, the quality of the individual sentences was vitiated. More concretely, the relevance between the sentence and the theme as well as the relevance between the sentences were lost. We except that this problem can be alleviated if the necessary information can be extracted from the previously generated sentence and passed to the next sentence, instead of the whole sentence. Investigation on how to design the input data in the sentence generation is the possible next research direction.

Automatic Evaluation of Essay

In the experiment in this thesis, the essays generated by the models were evaluated by human subjects. However, it would be better if we can evaluate essay fully automatically. It is not only for the evaluation of the final results but also for the parameter adjustment during the training of the models. In general, in order to optimize hyperparameters in deep learning, many models with different hyperparameters are trained, then the best model is chosen by evaluation of the models using a development data. If the models are manually evaluated for the parameter optimization, it is very time-consuming and requires a lot of human efforts. Therefore, the automatic evaluation of essays is essential for the optimization of hyperparameters. Since it is known that the parameter optimization much influences the performance of the deep learning, finding automatic evaluation enables us to improve the quality of the generated essays.

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[2] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[3] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. *CoRR*, abs/1612.08083, 2016.

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

[5] Brent Harrison, Christopher Purdy, and Mark O. Riedl. Toward automated story generation with markov chainmonte carlo methods and deep neural networks. In *The AIIDE-17 Workshop on Intelligent Narrative Technologies*, pages 191–197, 2017.

[6] Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. Story generation from sequence of independent short descriptions. *CoRR*, abs/1707.05501, 2017.

[7] The All Japan Magazine, Book Publisher's, and Editor's Association. https://www.ajpea.or.jp/information/20200124/index.html. (in Japanese).

[8] Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339, 2016.

[9] Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. *CoRR*, abs/1801.10198, 2018.

[10] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017.

[11] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[12] Lara J. Martin, Prithviraj Ammanabrolu, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. Event representations for automated story generation with deep neural nets. *CoRR*, abs/1706.01331, 2017.

[13] Neil McIntyre and Mirella Lapata. Learning to tell tales: A data-driven approach to story generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 217–225, 2009.

[14] Haitao Mi, Baskaran Sankaran, Zhiguo Wang, and Abe Ittycheriah. Coverage embedding models for neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 955–960, 2016.

[15] Ashutosh Modi, Tatjana Anikina, Simon Ostermann, and Manfred Pinkal. InScript: Narrative texts annotated with script information. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3485–3493, 2016.

[16] Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, 2018.

[17] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*, 2010.

[18] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, 2015.

[19] Baskaran Sankaran, Haitao Mi, Yaser Al-Onaizan, and Abe Ittycheriah. Temporal attention model for neural machine translation. *CoRR*, abs/1608.02927, 2016.

[20] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017.

[21] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1139–1147, 2013.

[22] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[23] Jun Suzuki and Masaaki Nagata. Cutting-off redundant repeating generations for neural abstractive summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 291–297, 2017.

[24] Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–85, 2016.

[25] Lilian Wanzare, Alessandra Zarcone, Stefan Thater, and Manfred Pinkal. Inducing script structure from crowdsourced event descriptions via semi-supervised clustering. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 1–11, 2017.

[26] Koki Yoneda, Soichiro Yokoyama, Tomohisa Yamashita, and Hidenori Kawamura. Development of automatic haiku generator using LSTM. In *The 32nd Annual Conference of the Japanese Society for Artificial Intelligence*, pages 1B2OS11b01–1B2OS11b01, 2018.

[27] Fangzhou Zhai, Vera Demberg, Pavel Shkadzko, Wei Shi, and Asad Sayeed. A hybrid model for globally coherent story generation. In *Proceedings of the Second Workshop on Storytelling*, pages 34–45, 2019.

[28] Shibo Zhang, Yun Sha, and Xiaojie Wang. Reviews analysis based on sentence and word relevance. In *2014 Seventh International Symposium on Computational Intelligence and Design*, pages 43–46, 2014.

[29] Xingxing Zhang and Mirella Lapata. Chinese poetry generation with recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 670–680, 2014.