Paper:

Natural Language Generation Using Monte Carlo Tree Search

Kaori Kumagai^{*1}**, Ichiro Kobayashi**^{*1}**, Daichi Mochihashi**^{*2}**, Hideki Asoh**^{*3}**,**
Tomoaki Nakamura^{*4}**, and Takayuki Nagai**^{*4}

^{*1}Advanced Sciences, Graduate School of Humanities and Sciences, Ochanomizu University
2-1-1 Ohtsuka, Bunkyo-ku, Tokyo 112-8610, Japan
E-mail: kumagai.kaori@is.ocha.ac.jp
^{*2}The Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa city, Tokyo 190-0014, Japan
E-mail: daichi@ism.ac.jp
^{*3}National Institute of Advanced Industrial Science and Technology (AIST)
1-1-1 Umezono, Tsukuba, Ibaraki 305-8560, Japan
E-mail: h.asoh@aist.go.jp
^{*4}The University of Electro-Communications
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan
E-mail: {tnakamura, tnagai}@ee.uec.ac.jp

[Received March 17, 2018; accepted July 31, 2018]

We propose a method of simulation-based natural language generation that accounts for both building a correct syntactic structure and reflecting the given situational information as input for the generated sentence. We employ the Monte Carlo tree search for this nontrivial search problem in simulation, using context-free grammar rules as search operators. We evaluated numerous generation results from two aspects: the appropriateness of sentence contents for the given input information and the sequence of words in a generated sentence. Furthermore, in order to realize an efficient search in simulation, we introduced procedures to unfold syntactic structures from words strongly related to the given situational information, and increased the probability of selecting those related words. Through a numbers of experiments, we confirmed that our method can effectively generate a sentence with various words and phrasings.

Keywords: natural language generation, Monte Carlo tree search, syntactic tree, context-free grammar

1. Introduction

People subconsciously produce utterances in daily life according to different situations. When a person encounters a situation in which a dog eats a piece of bread, he or she retrieves appropriate words and creates a natural sentence, retaining the dependent relationships among the words in proper order, to describe the situation. In addition, the words and phrases used in the descriptions are diverse. For example, when a person describes "a dog," it is sometimes called "a puppy," or "a big dog." This ability of natural language generation (NLG) in such situations will become essential for robots and conversational agents in the future.

However, this problem is intrinsically difficult because it is difficult to encode what to say into a sentence while ensuring its syntactic correctness and the possibility of a variety of words and phrases. In previous work [1], we proposed using Monte Carlo tree search (MCTS) [2, 3], a stochastic search algorithm for decision processes, to find an optimal solution in the decision space. Upon receiving the situational input, we built a search tree of possible syntactic trees to generate a sentence by selecting proper rules through numerous random simulations of possible outputs. We confirmed that our method could generate sentences with various words and phrasings while ensuring their syntactic correctness.

However, there was room for improvement. In this study, in order to realize more accurate and efficient sentence generation, we have improved three aspects. One is an improvement in feasible search in MCTS. To evaluate a generated sentence, in addition to evaluation of the *n*-gram language model, we adopt evaluation of the content of a sentence. The second improvement is to achieving efficient search. This is achieved through two steps: one is changing the search policy in MCTS. In our prior study, we used the left-most derivation to unfold a syntactic structure; however, in this study, we unfold the most important part in a sentence, i.e., the predicate. The second step is changing the sampling method: we set the probabilistic distribution considering the situational input and the *n*-gram language model. The third improvement is preparing lexicons considering their semantic characteristics, i.e., the semantics of given words as situational input. In concrete terms, in [1], the lexicons used in a generated sentence had to be prepared manually in advance, whereas we obtain lexicons whose meanings are similar to the lexicon of the situational input based on distributed semantics. In addition, words that co-occur are obtained by a topic model.

The paper is organized as follows. Section 2 describes our work with respect to the existing work. Section 3 presents the basic algorithm for sentence generation using MCTS. In Section 4, we introduce how to evaluate generated sentences for feasible search (Section 4.1), and then we explain setting the upper confidence bounds one (UCB1)-value for MCTS to converge all these evaluation values (Section 4.2). Section 5 describes the two means for achieving efficient search, and in Section 6, we explain how to prepare the lexical resources. In Section 7, we conduct the experiment, and finally, we draw conclusions in Section 8.

2. Related Studies

As for the nondeterministic approach to NLG, some studies view NLG as a planning problem. As an example of planning, there is a method called pipe-line generation [4], in which the contents to be generated are first decided by considering what the generated sentence expresses, and then the contents are translated into natural language, which corresponds to surface realization. To replace pipeline generation, there are methods that simultaneously plan the contents of a generated sentence and translate it to natural language expressions. CRISP [5] and PCRISP [6] have adopted this approach.

As another way to handle the indeterminate characteristics of NLG, Lemon [7, 8] and Rieser [9] modeled dialog as Markov decision processes (MDPs) and solved them by means of reinforcement learning [10].

Similar to our approach, STRUCT [11] and S-STRUCT [12] considered the NLG process as an MDP with a suitably defined reward function to achieve efficient sentence generation using an MCTS algorithm. However, these methods aim to generate a target sentence with a constraint on input information in terms of semantics, so we cannot expect that a variety of sentences could be generated by these methods. Contrary to these methods, our proposed method is able to generate a variety of sentences.

Another nondeterministic approach uses a neural language model [13], Wen et al. [14–16] used a Long Short-Term Memory generator, which can learn from unaligned data by concurrently optimizing sentence planning and surface realization using a simple cross-entropy training criterion, and easily achieves language variation by sampling from output candidates. However, this method predicts a word sequence only, and does not consider syntactic structures.

As another search-based algorithm that considers syntactic structures in generating a sentence, Liu et al. [17] proposed a syntactic linearization of given words using beam-search for an appropriate structure for a sentence. However, it treats the word order problem of generating sentences using only given words. Technically, their method employs a beam search with a predefined beamwidth. On the other hand, MCTS realizes an efficient search that does not restrict the search range in advance.

Moreover, Silver et al. [18] developed AlphaGo, which defeated a top-level professional Go player. They combined MCTS with a deep reinforcement learning framework, and provided MCTS with learning ability; both the policy and value networks of the system are trained to predict human expert behaviors using deep reinforcement learning. This framework could be applied to NLG in the future.

3. NLG with MCTS Simulations

3.1. MCTS

MCTS combines random simulation and best-first search in its search process [2]. It has been successfully applied as an algorithm for playing the Go game and similar planning problems. In fact, both the Go game and NLG share the same characteristic: their outputs can be evaluated only when the process reaches the last state. Therefore, we expect that the process of NLG can be represented in MCTS simulations.

MCTS uses the UCB1 value to determine the next move from the viewpoint of the multi-armed bandit problem [19].

$$\mathrm{UCB1} = v_i + C\sqrt{\frac{2\log N}{n_i}}. \quad \ldots\ldots\ldots \quad (1)$$

Here, v_i is the winning rate of candidate i, C is an adjustment coefficient, N is the total number of simulations, and n_i is the number of visits to the candidate i. The first term of Eq. (1) corresponds to exploitation and the second term corresponds to exploration within the simulation, achieving a balanced search between the two factors [20].

3.2. Algorithm

Our MCTS provides opportunities for selecting various syntactic structures and words for a generated sentence. We use probabilistic context-free grammar (PCFG) rules obtained from the Penn Tree Bank [21] as a search operator in MCTS.

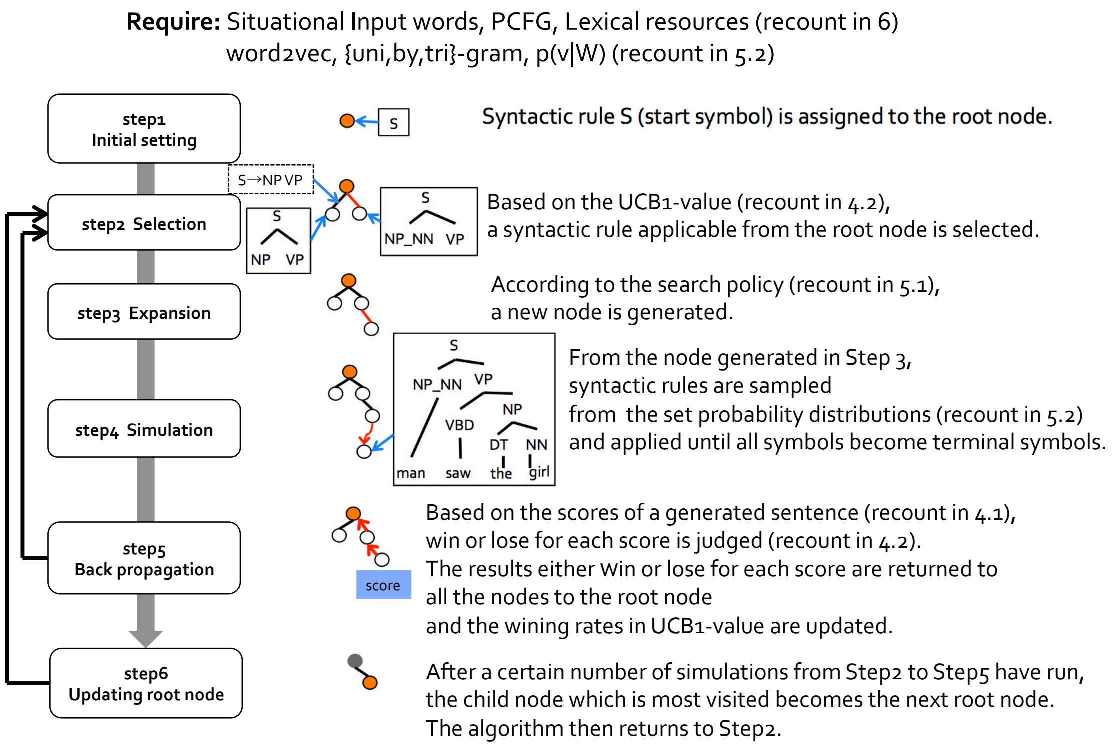
The MCTS algorithm is shown in **Fig. 1**.

Essentially, our MCTS builds a search space of possible derivations, and starting from the initial symbol S, we iteratively determine what rule to apply to extend the current tree, by simulating numerous possible derivations from the candidate rules.

In addition, as the content to press, we input situational information corresponding to the lexical information of subject, predicate, and object (SVO).

4. Method for Feasible Search

Unlike win or lose in the Go game, it is difficult for a machine to evaluate generated sentences in a binary state.


Require: Situational Input words, PCFG, Lexical resources (recount in 6)
word2vec, {uni,by,tri}-gram, p(v|W) (recount in 5.2)

Syntactic rule S (start symbol) is assigned to the root node.

Based on the UCB1-value (recount in 4.2),
a syntactic rule applicable from the root node is selected.

According to the search policy (recount in 5.1),
a new node is generated.

From the node generated in Step 3,
syntactic rules are sampled
from the set probability distributions (recount in 5.2)
and applied until all symbols become terminal symbols.

Based on the scores of a generated sentence (recount in 4.1),
win or lose for each score is judged (recount in 4.2).
The results either win or lose for each score are returned to
all the nodes to the root node
and the wining rates in UCB1-value are updated.

After a certain number of simulations from Step2 to Step5 have run,
the child node which is most visited becomes the next root node.
The algorithm then returns to Step2.

Fig. 1. MCTS algorithm for NLG.

In order to realize a feasible search, it is necessary to set up an appropriate evaluation for a generated sentence. In this section, we describe how to set evaluation values for a sentence, and how to set UCB1-values to converge all of these evaluation values.

4.1. Evaluation Values for Generated Sentences

We explain the score of the generated sentence described in step 5 in **Fig. 1**. As the score of a generated sentence for a feasible search, we adopt evaluation values from two perspectives: one is the correctness of the sequence of words, and the other is the appropriateness of the contents of a sentence. The details of the evaluation values are as follows:

1. Appropriateness of sentence contents
   We used distributed representations by Word2vec for situational input words, and the words used as SVO in a generated sentence, and used their cosine similarities as evaluation values.
2. Correctness of the sequence of words
   We used evaluation values based on the n-gram language model. Moreover, we used one of the following two types and compared the tendency of the words that appeared in the generated sentences.

   - *Perplexity* (hereinafter, *PP*)
     We use the perplexity of tri-grams with Kneser-Ney smoothing [22].
   - *Acceptability* (hereinafter, *AP*)
     We use a score called *Acceptability* proposed in [23], which measures the acceptability of a sentence for an English native speaker. In this study, we use *Acceptability* as defined by Eq. (2) for a sentence s:

$$Acceptability(s) = \log\left(\frac{p_{model}(s)}{p_{uni}(s)}\right)^{\frac{1}{|s|}} \qquad (2)$$

As an n-gram language model $p_{model}(s)$, we use trigrams with Kneser-Ney smoothing [22], where $p_{uni}(s)$ denotes the probability with a unigram distribution.

4.2. UCB1-Value and Judgement of Winning

We explain how to judge winning or losing at step 4 in **Fig. 1**. In order to converge all of the evaluation values mentioned in Section 4.1, the UCB1-value in this study is defined according to the following Eq. (3).

$$v_i^{cond.} + v_i^S + v_i^V + v_i^O + v_i^{PPorAP} + C\sqrt{\frac{2\log N}{n_i}} \quad . \qquad (3)$$

Here, because the expected values of the sum of the winning rates is $[0,5]$, we set C to 5 to maintain balance between exploitation and exploration in the simulation. We explain $v_i^{cond.}$, v_i^S, v_i^V, v_i^O, and v_i^{PPorAP} in the following.

- $v_i^{cond.}$
  First, we set the minimum conditions of a sentence so that the generated sentence should have SVO, and

satisfies the constraints on sentence length. When these conditions are satisfied, this state is regarded as a win. Then, 1 point is returned in the calculation of the winning rate. On the other hand, when the conditions are not satisfied, this state is regarded as a loss, then 0 is returned. The reason for setting the minimum conditions is explained in the following. In the initial search, we observed many sentences without SVO. Furthermore, there was difficulty in creating a difference in the UCB-1 value between candidate nodes, which slowed the convergence of the search. To accelerate the convergence of search at the initial stage, it was necessary to provide rewards when the minimum conditions for generating a sentence were satisfied.

- v_i^S, v_i^V, v_i^O
  These are the winning rates based on cosine similarity, which are regarded as evaluation values for the appropriateness of sentence contents mentioned previously. Only when the minimum conditions mentioned above are satisfied do we calculate the cosine similarities of S, V, and O. When those values are larger than their averages at other candidate nodes, a win (1 point) is returned, and otherwise 0 is returned.
- v_i^{PP} or v_i^{AP}
  This is the winning rate using AP or PP, and is regarded as an evaluation value for the correctness of the sequence of words mentioned previously. Only when v_i^S, v_i^V, and v_i^O win do we calculate either PP or AP. Moreover, when the value is larger than PP or AP averages at other candidate nodes, 1 is returned as a winning point, and otherwise 0 is returned.

5. Method for Efficient Search

In this section, we explain two approaches for efficient search.

5.1. Search Policy and SVO Judgment

In the previous work [1], we used left-most derivation in the formation of a syntactic tree. In this case, a word is selected depending on the confirmed left-most word. For example, if the left-most word is the article "a," the word to the right of "a" has to be selected considering the connection with "a." In this paper, a syntactic tree is built from elements strongly related to situational input. Considering this, a syntactic tree is built in the following order: verb, noun, either adjective or adverb, and stop words. The reason for building from a verb first is that the predicate is related to the lexical selection of both the subject and the object. Here, because SVO is not directly judged by the syntactic tree built by PCFG, a syntactic tree is built in the following order, and then SVO is determined.

1. Unfold the part-of-speech (POS) related to verb
   The terminal symbol initially unfolded is regarded as the predicate of a generated sentence.
2. From the left side of predicate, unfold the POS related to noun
   The terminal symbol initially unfolded is regarded as the subject of a generated sentence.
3. From the right side of predicate, unfold the POS related to noun
   The terminal symbol initially unfolded is regarded as the object of a generated sentence.
4. Unfold the POS related to either adjective or adverb.
5. Unfold the other POSs.

In 1, 2, and 3 above, the reason why the initially unfolded terminals should be SVO is because there is a high possibility that the words located in the shallow part of a syntactic tree become important components in a sentence.

5.2. Setting Probabilistic Distribution for Sampling Target

In the simulation (step 3 in **Fig. 1**), in the previous work [1], we sampled the grammar rules from a uniform distribution. In this case, for example, even if "eat" is given as the predicate of the situational input, "eat" and "run" are selected with the same probability. In order to achieve an efficient simulation, we set probability distributions for sampling grammar rules, considering situational input, connection with surrounding words, and PCFG probabilities. Specifically, we set probability distributions by choosing nonterminal symbols, SVO, independent words not including the words for SVO, and stopwords. The details are shown below.

- Nonterminal symbols
  We use the probabilities from PCFG.
- SVO
  In order to increase the probability of selecting words whose meanings are similar to the words in the situational input, we use distributed representations by Word2vec and adapt the cosine similarity to each situational input word. The normalization method for cosine similarity is shown in Eq. (4).

$$\frac{\exp(\beta r)}{\sum(\exp(\beta r))} \quad \ldots\ldots\ldots\ldots (4)$$

  Here, r is cosine similarity, and $\beta = 2.0$.
- Independent words not including SVO
  In order to increase the probability of selecting words that are likely to co-occur with situational input words, we obtain the word distributions $p(v|W)$, where W indicates situational input words, e.g., {dog, eat, bread}. We use latent Dirichlet allocation (hereinafter, called LDA) [24]. First, we find the topic distribution θ_{new} on the new document W using the pre-trained LDA model – let this be $p(\theta_{new}|W)$. Second, using the word distribution for each topic ϕ, we obtain $p(v|\theta_{new})$ from Eq. (5). Finally, $p(v|W)$ is obtained from Eq. (6).

$$p(v|\theta_{new}) = \sum_k \phi_{k,v}\theta_{new,k} \quad \ldots\ldots\ldots (5)$$

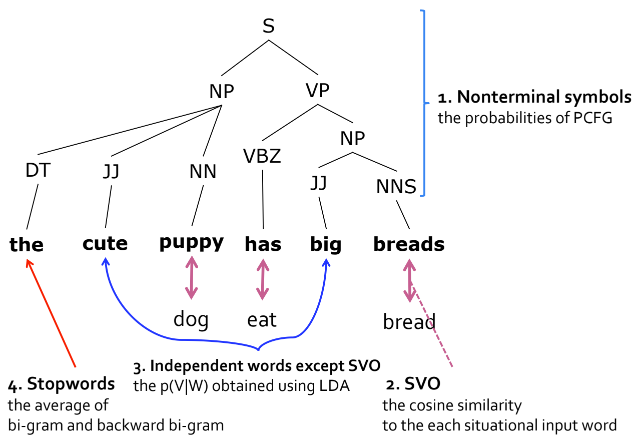

S
NP
VP
VBZ
NP
DT
JJ
NN
JJ
NNS
the
cute
puppy
has
big
breads
dog
eat
bread
1. Nonterminal symbols
the probabilities of PCFG
2. SVO
the cosine similarity
to the each situational input word
3. Independent words except SVO
the p(V|W) obtained using LDA
4. Stopwords
the average of
bi-gram and backward bi-gram

Fig. 2. Overview of setting of probabilistic distribution.

$$p(v|W) = \int p(v|\theta_{new})p(\theta_{new}|W)d\theta_{new} \quad . \quad (6)$$

- Stopwords
  Considering the connectivity of the words before and after, we obtain the bigram and backward bigram probabilities and use their average value.

An image of the above explanation for this method of choosing words is shown in **Fig. 2**.

6. Lexical Resources

In the case of using MCTS to generate a sentence, if there are quite a few lexical elements collected from the corpus, it enlarges the search space for the algorithm. This becomes a considerable problem if a large number of simulations are required. In order to solve this problem, it is necessary to pare down lexical resources appropriately based on the situational input. In the previous work [1], when we narrowed down lexicons that we likely to co-occur with the situational input, we collected lexical resources by collecting two words before and after each word of situational input from the corpus. This method has to be done manually before the MCTS simulation. In any case, the situational input words cannot all be considered simultaneously. That is, in the case that situational input words are "dog," "eat," and "bread," the lexical resources become the combination of words likely to co-occur with "dog," "eat," and "bread." In this paper, considering the collection of lexicons that are likely to co-occur simultaneously with all the words of situational input, we adopt a probability distribution of words obtained by a topic model. The concrete procedure of narrowing down lexicons is shown below.

1. Delete the lexicons whose number of occurrences in the corpus is less than 5.
2. Consider the distribution we have set in Section 5.2.

- SVO
  Delete the lexicons whose cosine similarity is less than 0.3.
- Independent words other than SVO
  In each part-of-speech, delete the lexicons except the ones in the top 20 based on the conditional probability $p(v|W)$ mentioned in Section 5.2.

7. Experiment

7.1. Experimental Settings

As for the corpus for LDA, we used all 85,413 sentences in the Microsoft Research Video Description Corpus (MVDC).[1] We also used Wikipedia English Corpus[2] to make Word2vec distributed semantics and *n*-grams. As for making PCFG, we used the corpus 0000–0009 of WST100 in Penn TreeBank [21], and made 717 syntactic rules from the corpus. As for lexicons, we used 4,464 words that appear more than 5 times in MVDC. As the situational input to generate the contents of a sentence, we evaluated three kinds of situational input as shown below.

$$W = \{dog, eat, bread\}, \{boy, play, soccer\} \\ \{man, write, letter\}$$

We conducted experiments with two sets of constraints on the length of a generated sentence as it becomes three to five or six to eight words in length. The number of simulations is set to 100 for the number of the candidate nodes.

7.2. Results and Discussion

The result of sentence generation is shown in **Table 1**.

As shown in **Table 1**, various expressions of a sentence are generated under the same conditions. As for SVO, we confirmed that proper words were selected for those categories. As we can see, there is a case where *read* was selected when *write* was provided for V, we have confirmed that there is a case where various words with large cosine similarity in the embedding space by Word2vec were selected. We also confirmed that various independent words such as nouns, adjuncts, and verbs appeared when using *AP* rather than *PP*.

Next, we analyze the evaluation values of the sentences among searching. **Fig. 3** shows the transitions in the ratio of satisfying the minimum conditions of a sentence each time the root node is updated with an example of generating a sentence with the situational input {dog, eat, bread} in MCTS simulation. We found that this probability converges to 1 as the search progresses. That is, at the beginning of the search it is difficult to satisfy the minimum conditions. On the other hand, in the latter half of the search, the conditions are satisfied in most cases.

Figure 4 shows the transitions of the average and variance for each evaluation value for each time the root node

1. https://www.microsoft.com/en-us/download/details.aspx?id=52422
2. http://dumps.wikimedia.org/enwiki/

Table 1. Examples of generated sentence.

W	*Len.*	*AP*	*PP*
$\{dog, eat, bread\}$	$3\sim5$	· another dog eaten smallest breads	· both dog eats quick bread
		· another dog ate connecticut breads	· both dog gather bread washington
		· another dog eats smallest breads	· dog eating eight bread
		· another dog eating smallest breads	· dog ate 1 bread
		· dogs eaten bread connecticut	· the dog eaten either breads
	$6\sim8$	· either dog ate underneath jeff connecticut breads	· both dog ate with the breads and passengers
		· another dog eaten automatically recumbent breads automatically	· both dog eat on combining breads
		· every dog eat underneath another outstretched bread breads	· a dog eat with italian quick breads poles
		· another dog eats automatically recumbent breads	· the dog eaten bread ’ aggressively
		· another dog eats automatically battering breads	· an dog eat lighter breads overboard
$\{boy, play, basketball\}$	$3\sim5$	· another boy play overweight soccer	· refreshing boys played soccer
		· another boy play eight soccer	· the boy play an soccer
		· boys play another soccer ’s	· powerful boys played soccer
		· another boy played overweight soccer	· every boy play five soccer
		· another boy play eight soccer	· the boy play their soccer
	$6\sim8$	· another boy play underneath soccer vaccination	· any boy play for any soccer
		· half boy play underneath soccer connecticut	· every boy play till olympic soccer
		· these boy play underneath soccer vaccination	· a boy play between olympic soccer
		· another boy play recumbent soccer horizontally fistfighting soccer	· the boy play soccer episode ritually
		· the boy play underneath soccer vaccination proffestionals	· any boy play for its soccer
$\{man, write, letter\}$	$3\sim5$	· man learn letter vaccination	· an man learn 10 letter
		· another man read letter	· these man read letter washington
		· overweight men read letter	· men read both italian letter
		· men wrote letter vaccination	· man learn your letter
		· man wrote another smallest letter	· these woman wrote letter washington
	$6\sim8$	· that man wrote underneath letter connecticut	· the man read in no smallest letter
		· another man read together letter connecticut	· the man read open letter episode
		· another man read together letter vaccination letters binoculars	· the man read together italian letter
		· overweight men read another overweight letter	· the man read in both letter ’
		· another man read together letter connecticut	· the man read in an lower letter

is updated in the MCTS simulation in the above example. From this figure, we can see the variances of all evaluation values converge to low values. Although there are no big changes, the averages are slightly higher at the end of the search than at the beginning. In other words, it means the search progressed so that all the values converged at high values.

Next, in comparison with the previous work [1], we consider the effects obtained by the three improvements. First, as for the effect of the method for feasible search, because the generated sentence is evaluated for the meaning of the words for SVO, words that are similar to situational input words are used for SVO in the sentence. Therefore, it is possible to generate sentences considering the syntactic categories of situational input words. For example, in the previous work [1], there is a possibility that a word corresponding to the situational input may not be assigned to the position of SVO. In fact, when the situational input words are {boy, play, basketball}, the sentence “boys tennis was played to all the” was generated. On the contrary, in the current method, we confirmed that words corresponding to situational input words were assigned to the position of SVO of the generated sentence. Second, we explain the effect obtained by the method for

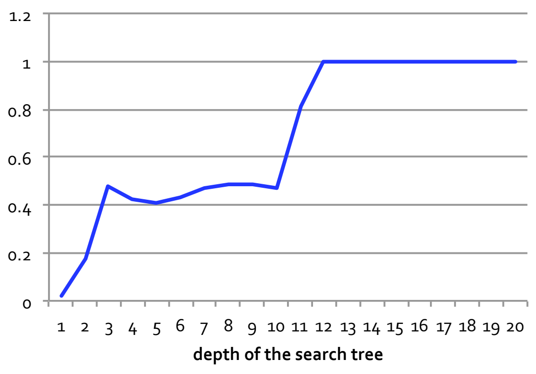


Fig. 3. Ratio of satisfying the minimum conditions (sentence of length 6 to 8 words, and having SVO).

efficient search. In the generation process, we consider syntactic categories, i.e., SVO, in the unfolding order of a syntactic tree, as well as the probabilistic distribution of syntactic rules, therefore, syntax errors are reduced. For example, in the above example sentence “boys tennis was played to all the,” not only is the content of the sentence

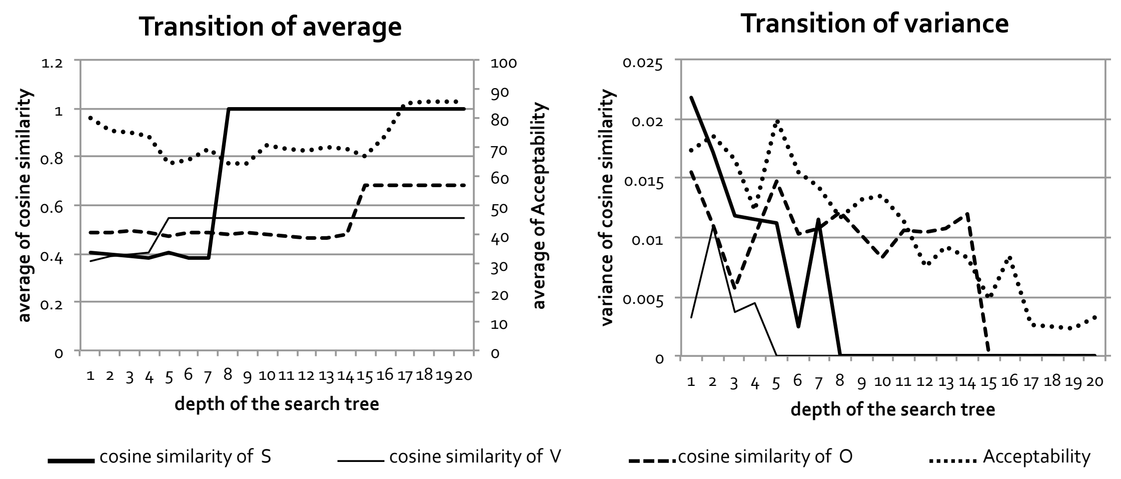


Fig. 4. Transitions of average and variance for each evaluation value.

wrong, but errors in syntactic structure can also be seen. Third, we describe the effects of changing the method of paring down lexical resources. In the previous work [1], we collected words appearing with the words given as situational input words. In contrast, in this method, because we selected words that were likely to co-occur with all situational input words by means of a topic model, we could narrow down words efficiently. Specifically, when {dog, eat, bread} were received as situational input words, in the previous work [1], there were 20798 words. On the other hand, there were only 215 words by this method.

Next, we will refer to the grammatical errors seen in the generated sentences. The purpose is to generate sentences with correct syntactic structures and a variety of words and phrasings, so we did not consider grammatical information such as verbs and noun deformations. For this reason, grammatical errors such as transformation of verbs by tense and change of inflection of single and plural nouns are seen in the generated sentences. On the other hand, because the methods use neural networks, which have often been referred to in recent years as a sentence generation method, use language models that are learned from a large amount of data, they can cope with the deformation of verbs and noun statistically. However, because words are selected based on statistical information, the model is influenced by frequent words in the training data. Thus, it is difficult to generate a variety of sentences, and the methods using neural networks can not achieve our purpose.

Finally, we describe the problem in this experiment. We observed the generation of sentences that were semantically difficult to understand. This may be the reason why we did not provide any information about the words except the words for SVO. We think that the evaluation for a generated sentence is still insufficient to some extent.

8. Conclusion

In this study, we proposed a robust method to generate sentences with various phrasing for the same meaning, using MCTS and applying PCFG syntactic rules as search operators. To effectively achieve MCTS, we devised the settings for search, the way of selecting words, the evaluation metric for generated sentences, the lexical resources, UCB1 value, etc. In the experiments, we confirmed that various sentences were generated by following the input information, however, there were cases where uninterpretable sentences were generated. In the future, we will revise the evaluation metric for generated sentences and then generate more understandable sentences.

Acknowledgements

Part of this study was financially supported by KAKENHI (15K12104).

References:

[1] K. Kumagai, I. Kobayashi, D. Mochihashi, H. Asoh, T. Nakamura, and T. Nagai, "Human-like Natural Language Generation Using Monte Carlo Tree Search," Proc. of the INLG 2016 Workshop on Computational Creativity in Natural Language Generation, pp. 11-18, 2016.

[2] L. Kocsis and C. Szepesvari, "Bandit based Monte Carlo planning," ECML 2006, pp. 282-293, 2006.

[3] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "Survey of Monte Carlo Tree Search Methods," Technical Report 1, 2012.

[4] E. Reiter and R. Dale, "Building applied natural language generation systems," Nat. Lang. Eng., Vol.3, No.1, pp. 57-87, 1997.

[5] A. Koller and M. Stone, "Sentence generation as a planning problem," Int. Natural Language Generation Workshop, Vol.12, pp. 17-24, 2007.

[6] D. Bauer and A. Koller, "Sentence generation as planning with probabilistic LTAG," Proc. of the 10th Int. Workshop on Tree Adjoining Grammar and Related Formalisms, 2010.

[7] O. Lemon, "Adaptive natural language generation in dialogue using reinforcement learning," Workshop on the Semantics and Pragmatics of Dialogue (SEMDIAL), pp. 141-148, 2008.

[8] O. Lemon, "Learning what to say and how to say it: joint optimization of spoken dialogue management and natural language generation," Computer Speech and Language, Vol.25, No.2, pp. 210-221, 2011.

[9] V. Rieser and O. Lemon, "Natural Language Generation as Planning under Uncertainty for Spoken Dialogue Systems," EACL 2009, pp. 683-691, 2009.

[10] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," MIT Press, 1998.

[11] N. McKinley and S. Ray, "A Decision-Theoretic Approach to Natural Language Generation," Proc. of the 52nd Annual Meeting of the Association for Computational Linguistics, Vol.1: Long Papers, pp. 552-561, 2014.

[12] J. Pfeil and S. Ray, "Scaling a Natural Language Generation System," Proc. of the 54th Annual Meeting of the Association for Computational Linguistics, Vol.1: Long Papers, pp. 1148-1157, 2016.

[13] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A Neural Probabilistic Language Model," J. of Machine Learning Research, Vol.3, pp. 1137-1155, 2003.

[14] T.-H. Wen, M. Gašić, D. Kim, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young, "Stochastic Language Generation in Dialogue using Recurrent Neural Networks with Convolutional Sentence Reranking," Proc. of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), 2015.

[15] T.-H. Wen, M. Gašić, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young, "Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems," Proc. of the 2015 Conf. on Empirical Methods in Natural Language Processing (EMNLP), 2015.

[16] T.-H. Wen, M. Gašić, N. Mrkšić, L. M. Rojas-Barahona, P.-H. Su, D. Vandyke, and S. Young, "Multi-domain Neural Network Language Generation for Spoken Dialogue Systems," Proc. of the 2016 Conf. on North American Chapter of the Association for Computational Linguistics (NAACL), 2016.

[17] Y. Liu, Y. Zhang, W. Che, and B. Qin, "Transition-Based Syntactic Linearization," NAACL 2015, pp. 113-122, 2015.

[18] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," Nature, Vol.529, pp. 484-489, 2016.

[19] M. N. Katehakis and A. F. Veinott, "The Multi-Armed Bandit Problem: Decomposition and Computation," Mathematics of Operations Research, Vol.12, No.2, pp. 262-268, 1987.

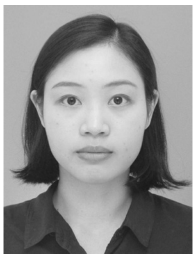
[20] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multi-armed bandit problem," Machine Learning, Vol.47, pp. 235-256, 2002.

[21] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a Large Annotated Corpus of English: The Penn Treebank," Computational Linguistics, Vol.19, No.2, pp. 313-330, 1993.

[22] R. Kneser and H. Ney, "Improved backing-off for n-gram language modeling," ICASSP, Vol.1, pp. 181-184, 1995.

[23] J. H. Lau, A. Clark, and S. Lappin, "Unsupervised Prediction of Acceptability Judgements," ACL 2015, Vol.53, pp. 15-1000, 2015.

[24] D. Blei, A. Ng, and M. Jordan, "Latent Dirichlet Allocation," J. of Machine Learning Research, Vol.3, pp. 993-1022, 2003.



Name:
Kaori Kumagai

Affiliation:
Advanced Sciences, Graduate School of Humanities and Sciences, Ochanomizu University

Address:
2-1-1 Ohtsuka, Bunkyo-ku, Tokyo 112-8610, Japan

Brief Biographical History:
2017 Received Master Degree in Information Science, Ochanomizu University
2017- Doctor Candidate, Information Science, Ochanomizu University

Main Works:
• "Human-like Natural Language Generation Using Monte Carlo Tree Search," Proc., INLG 2016 Workshop on Computational Creativity in Natural Language Generation, pp. 11-18, 2016.

Name:
Ichiro Kobayashi

Affiliation:
Advanced Sciences, Graduate School of Humanities and Sciences, Ochanomizu University

Address:
2-1-1 Ohtsuka, Bunkyo-ku, Tokyo 112-8610, Japan

Brief Biographical History:
1995 Assistant Professor, Hosei University
1996-2003 Associate Professor, Hosei University
2003-2010 Associate Professor, Ochanomizu University
2011- Professor, Ochanomizu University
2017- Visiting Research Scholor, National Institute of Advanced Industrial Science and Technology (AIST)

Main Works:
• "Describing Semantic Representations of Brain Activity Evoked by Visual Stimuli," IEEE-SMC2018.

Membership in Academic Societies:
• The Association for Computational Linguistics (ACL)
• The Japanese Society for Artificial Intelligence (JSAI)

Name:
Daichi Mochihashi

Affiliation:
The Institute of Statistical Mathematics

Address:
10-3 Midori-cho, Tachikawa city, Tokyo 190-0014, Japan

Brief Biographical History:
1998 B.S. Degree, the University of Tokyo
2005 Ph.D., Nara Institute of Science and Technology
2003-2006 ATR Spoken Language Communication Research Laboratories
2007-2010 NTT Communication Science Laboratories
2011- The Institute of Statistical Mathematics

Main Works:
• R. Fujii, R. Domoto, and D. Mochihashi, "Nonparametric Bayesian Semi-supervised Word Segmentation," TACL, Vol.5, pp. 179-189, 2017
• T. Nakamura et al., "Segmenting Continuous Motions with Hidden Semi-Markov Models and Gaussian Processes," Frontiers in Neurorobotics, Vol.11, p. 67, 2017.

Membership in Academic Societies:
• Information Processing Society of Japan (IPSJ)
• The Japan Statistical Society

Name:
Hideki Asoh

Affiliation:
Deputy Director, Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology (AIST)

Address:
1-1-1 Umezono, Tsukuba, Ibaraki 305-8560, Japan

Brief Biographical History:
1983- Electrotechnical Laboratory, Ministry of International Trade and Industry
1993-1994 Visiting Researcher, German National Research Center for Information Technology (GMD)
2001- National Institute of Advanced Industrial Science and Technology (AIST)

Main Works:
• "Jijo-2: An office robot that communicates and learns," IEEE Intelligent Systems, Vol.16, No.5, pp. 46-55, 2001
• "On the mean convergence time of evolutionary algorithms without selection and mutation," Proc. of the 3rd Int. Conf. on Parallel Problem Solving from Nature, pp. 88-97, 1994.

Membership in Academic Societies:
• The Japanese Society for Artificial Intelligence (JSAI)
• The Institute of Electronics, Information and Communication Engineers (IEICE)
• Japanese Neural Network Society (JNNS)

Name:
Tomoaki Nakamura

Affiliation:
Department of Mechanical Engineering and Intelligent Systems, The University of Electro-Communications

Address:
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan

Brief Biographical History:
2011- Research Fellow, Japan Society for the Promotion of Science
2013- Researcher, Honda Research Institute Japan Co., Ltd.
2014- Assistant Professor, The University of Electro-Communications

Main Works:
• "Online Algorithm for Robots to Learn Object Concepts and Language Model," IEEE Trans. on Cognitive and Developmental Systems, Vol.9, No.3, pp. 255-268, 2017.

Membership in Academic Societies:
• The Robotics Society of Japan (RSJ)
• The Japanese Society for Artificial Intelligence (JSAI)

Name:
Takayuki Nagai

Affiliation:
Professor, Graduate School of Informatics and Engineering, The University of Electro-Communications

Address:
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan

Brief Biographical History:
1998- Associate Professor, The University of Electro-Communications
2002- Visiting Researcher, University of California, San Diego
2014- Professor, The University of Electro-Communications

Main Works:
• "Online Algorithm for Robots to Learn Object Concepts and Language Model," IEEE Trans. on Cognitive and Developmental Systems, Vol.9, No.3, pp. 255-268, 2017.

Membership in Academic Societies:
• The Robotics Society of Japan (RSJ)
• The Japanese Society for Artificial Intelligence (JSAI)
• The Institute of Electrical and Electronics Engineers (IEEE)