User's Intention Understanding in Question-Answering System Using Attention-based LSTM

Yûki Matsuyoshi* and Tetsuya Takiguchi† and Yasuo Ariki‡
Graduate School of System Informatics, Kobe University, Kobe, Japan
E-mail: *matsuyoshi@me.cs.scitec.kobe-u.ac.jp, †takigu@kobe-u.ac.jp, ‡ariki@kobe-u.ac.jp

***Abstract*—A rule-based question-answering system is limited in its ability to understand a user's intention due to the inevitable incompleteness of the rules. To address this problem, in this paper, we propose a method to estimate *question type* and *question keyword class* from a user's question by using an *attention-based LSTM (Long Short-Term Memory)* model. We also propose a *joint model* for simultaneous estimation of *question type* and *question keyword class*. Through the experiment, the effectiveness of our proposed method is evaluated based upon estimation rates. In addition, the proposed method for *question type* estimation is compared with a rule-based system, support vector machine (SVM), and Random Forest. The method for *question keyword class* estimation is also compared with the *non-attention LSTM* model and the conventional model.**

I. Introduction

In an information society, we have many opportunities to use machines and computer software in various fields. We read the operation manuals at the beginning before using such technology, but it is difficult for us to get accustomed to using them. From this view point, in this study, we aim to construct a system that can help users to understand its usage by answering their questions interactively during the operations. As a preliminary step, an interactive support system is constructed for Othello games, because in board games, such as Othello, players will improve their skills and understand the games through reading rule books, playing, and teaching each other.

Typically, a spoken language understanding system performs intent detection and slot filling to extract the speaker's intention and semantic constituents from the speaker's utterance. Intent detection can be thought of as an utterance classification problem, and slot filling as a sequence-labeling problem. In intent detection, SVM [1], and deep neural networks [2] are often used. In slot filling, recurrecnt neural networks (RNN) [3], and convolutional neural networks (CNN) [4] are often used. Recently, encoder-decoder neural network models are being used in slot filling [5]. In addition, the *attention mechanism* [6] enables the encoder-decoder model to learn aligning and decoding simultaneously.

We have already constructed a rule-based question answering system, which was composed of a *question analysis unit* and *response generation unit*. In the *question analysis unit*, *question type* and *question keyword class* are estimated, as an intention of a user's question, from an input sentence using keyword spotting. The number of *question type*s is 5, and the number of *question keyword classes* is 21. Then *question type* and *question keyword class* are sent to the *response generation unit* together with current board parameters from the Othello program. In the *response generation unit*, answers are generated by rules that consist of *question type*, *question keyword class*, parameters from the Othello program, and answer templates. We set a total of 202 rules. They fill in the answer template with the value of *question keyword class* and parameters from the Othello game. There are various problems in the constructed rule-based system. The most fatal problem involves handling the diversity of user's questions by if-then rules. In addition, if the system is applied to a domain other than Othello games, new rules have to be created for the domain. In order to solve these problems, we thought it neccesary for the system to be able to flexibly analyze the variable questions automatically.

In this work, assuming that the interactive support is in the form of *question and answering*, we estimate a user's intention from their questions by identifying *question type* and *question keyword class*. *Question type* estimation can be treated as intent detection, and *question keyword class* estimation can be treated as slot filling. We propose the *attention-based Long Short-Term Memory (LSTM)* model for *question type* estimation and the *attention-based LSTM encoder-decoder* model for *question keyword class* estimation. Unlike the conventional *attention-based LSTM encoder-decoder* model [7,8], our proposed model uses not only the output from the hidden layer at one previous time step but also uses the outputs from the hidden layers at two previous time steps as the input to the decoder. In addition, we propose a *joint model* which combines the estimation model of *question type* and the estimation model of *question keyword class* and performs them simultaneously.

II. Proposed System

A. Outline of the System

The outline of our system is shown in Fig. 1. A User's question is first sent to the *question analysis unit*. Then, using the results of the question analysis and parameters from the Othello program, as well as information in the knowledge database about Othello, the *inference engine* generates an optimum answer for the user. Also, if the question sentence is judged to be *chat*, the *chat system* generates an answer for

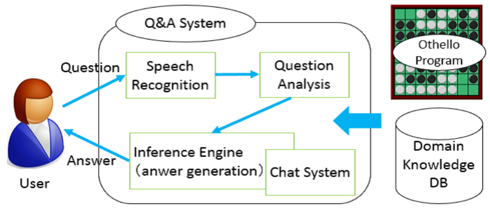

Fig. 1. The outline of our proposed system.

the user. In this paper, we focus on the construction of the *question analysis unit*.

B. Question Analysis Unit

Since we deal with the Japanese language in this study, we need to decompose a user's question sentences into morphemes. For this purpose, the Japanese morphological analyzer *Mecab* [9] is used to decompose the question sentence into morphemes. Consequently the question sentence data is converted to a morpheme string.

Question type and *question keyword class* are estimated from morpheme strings of a user's question. *Question type* expresses an outline of questions and is a user intention. We prepare 15 *question types* in total. Examples are shown below.

`Reason`: Questions about the reason to system's answer.
`Location`: Questions about locations on a board.
`Chat`: Daily conversation other than questions about Othello.

Question keyword class is higher-order terms of keywords appearing in a question sentence, and we prepare 14 kinds in total. Examples are shown below.

`Term`: Othello technical terms such as *X-square*, *Liberty*.
`Coordinate`: Coordinates of Othello board such as b8.

C. Question Type Estimation

The *question type* is estimated using the *attention-based bidirectional LSTM* [8] which is the LSTM model with the *attention mechanism* [2] as shown in Fig. 2. First, each morpheme x_i(i=1,2,...,m) in a question sentence is converted into a one-hot word vector. Then, the word vector is transformed into a distributed representation by word embedding, and fed to the LSTM in chronological order. The forward and backward hidden layers $\boldsymbol{h_{fi}}$ and $\boldsymbol{h_{bi}}$ of the LSTMs are concatenated into $\boldsymbol{h_i}$, using training parameters $\boldsymbol{W_1}$, $\boldsymbol{W_2}$:

$$\boldsymbol{h_i} = \boldsymbol{W_1 h_{fi}} \oplus \boldsymbol{W_2 h_{bi}} (i = 1, 2, ..., m) \tag{1}$$

When the end-of sentence symbol <eos> is given at the end of the input sentence [8], we assume the output of the hidden layer of LSTM is $\boldsymbol{h_o}$. The *attention* weight α_i is computed between the hidden layer $\boldsymbol{h_i}$(i=1,2,...,m) and $\boldsymbol{h_o}$:

$$\alpha_i = \frac{\exp(\boldsymbol{W_3^T} tanh(\boldsymbol{W_4 h_i} + \boldsymbol{W_5 h_o}))}{\sum_{j=1}^{m} \exp(\boldsymbol{W_3^T} tanh(\boldsymbol{W_4 h_j} + \boldsymbol{W_5 h_o}))} \tag{2}$$

$\boldsymbol{W_3}$, $\boldsymbol{W_4}$ and $\boldsymbol{W_5}$ are training parameters. Regarding α_i as the *attention* weight of $\boldsymbol{h_i}$, a context vector $\boldsymbol{c}$ is computed :

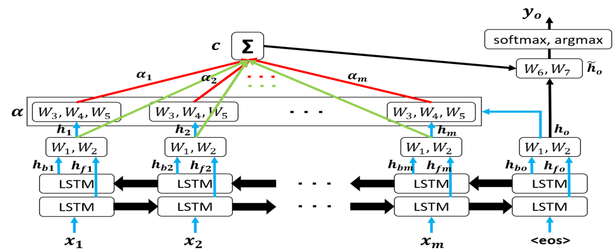

Fig. 2. The *attention-based bidirectional LSTM* model for *question type* estimation. Because, the last state of the forward and backward layer where <eos> is entered has information of the entire input sentence, it is useful for estimating *question type*.

$$\boldsymbol{c} = \sum_{i=1}^{m} \alpha_i \boldsymbol{h_i} \tag{3}$$

The output vector $\widetilde{\boldsymbol{h_o}}$ is calculated using $\boldsymbol{c}$ and $\boldsymbol{h_o}$:

$$\widetilde{\boldsymbol{h_o}} = tanh(\boldsymbol{W_6 c} + \boldsymbol{W_7 h_o}) \tag{4}$$

$\boldsymbol{W_6}$ and $\boldsymbol{W_7}$ are training parameters. The size of $\widetilde{\boldsymbol{h_o}}$ is converted to the size of the output vocabulary (the number of *question types*, 15 dimensions). After softmax operation to $\boldsymbol{h_o}$, a maximum value is selected as the estimate of the *question type* y_o :

$$y_o = argmax(softmax(\widetilde{\boldsymbol{h_o}})) \tag{5}$$

D. Question Keyword Class Estimation

Question keyword class estimation can be treated as slot filling. We train the models to learn a function that maps an input sequence to corresponding label sequence. therefore, the input morpheme sequence of the question sentence and the label sequence are of the same length [5].

Question keyword class is estimated by the *attention-based LSTM encoder-decoder* [7,8] which introduced the *attention mechanism* to a conventional LSTM encoder-decoder [4], as shown in Fig. 3. The encoder is similar to an *attention-based bidirectional LSTM* as described in II-C. The decoder is trained so that *question keyword class* (y_1, y_2, ...) are generated sequentially after <eos> is given.

In the conventional LSTM encoder-decoder [4,7], the encoder reads an input sequence x_i(i=1,2,...,m) and compresses it into the hidden layer $\boldsymbol{h_o}$, which has information for the whole input sequence and is used in the decoder to generate the output sequence $\boldsymbol{y}$. Context vectors $\boldsymbol{c}$ are calculated at each time step of outputs in the decoder. The decoder calculates the probability of the output sequence $\boldsymbol{y}$ as follows.

$$\begin{aligned} P(\boldsymbol{y} \mid \boldsymbol{h_o}) &= \prod_{t=1}^{T} P(y_t \mid y_1^{t-1}, \boldsymbol{h_o}) \\ &\fallingdotseq \prod_{t=1}^{T} P(y_t \mid y_{t-1}, \boldsymbol{h_o}) \end{aligned} \tag{6}$$

where y_{t-1} is the output from the hidden layer at one previous time step and y_1^{t-1} describes y_1,y_2,...,y_{t-1}. In our proposed model, the decoder also uses the outputs from the hidden

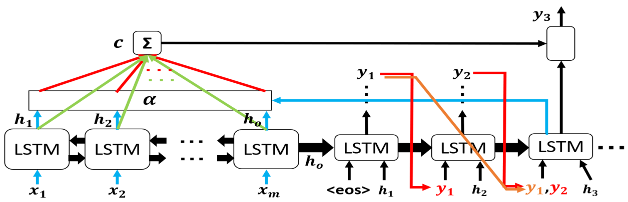

Fig. 3. The *attention-based LSTM encoder-decoder* model for *question keyword class* estimation. The encoder is a bidirectional LSTM, and the decoder is a unidirectional LSTM.

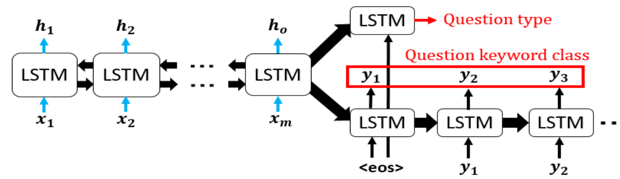

Fig. 4. The *joint model*. In this model, the *question type* estimation model and the *question keyword class* estimation model share the encoder and the *attention mechanism*.

layers at two previous time steps y_{t-2}. In the sequence labeling, since conditional probability can be improved by taking the long sentence as the condition, the following equation (7) contributes to improving conditional probability compared with equation (6).

$$P(\boldsymbol{y} \mid \boldsymbol{h_o}) = \prod_{t=1}^{T} P(y_t \mid y_1^{t-1}, \boldsymbol{h_o}) \fallingdotseq \prod_{t=1}^{T} P(y_t \mid y_{t-1}, y_{t-2}, \boldsymbol{h_o}) \qquad (7)$$

E. Joint Model Estimation

A *joint model* for intent detection and slot filling is proposed in spoken understanding in [7,10]. In such models, intent detection and slot filling can be learned by only one model. In this study, we built the *joint model* combining the *question type* estimation model described in II-C and the *question keyword class* estimation model described in II-D, sharing the same encoder as shown in Fig. 4. During the model training, the costs from both decoders are back propagated to the encoder. The *question type* estimation decoder generates a single output, which is the *question type* distribution, and the question keyword class estimation decoder generates sequential output, which are *question keyword class* estimation distributions. The *question type* estimation decoder and the question keyword class estimation decoder share last encoder state $\boldsymbol{h_o}$, which encodes information of the entire question sentence. Although omitted in Fig. 4, an *attention mechanism* in the encoder is also introduced, and it is shared in both the *question type* estimation decoder and the *question keyword class* estimation decoder.

III. Experiments

A. Training Details

We trained our models on a user's question corpus collected while playing the Othello game. The corpus consists of a triple group (one question sentence, one *question type*, *question keyword classes*). *Question type* and *question keyword class* were given as annotations to these question sentence data manually. The total number of data in the user's question corpus is 1,000, of which 895 were used for training and the remaining 105 were used for testing.

The distributions of *question type* and *question keyword class* in the training data set are shown in Table I and Table II. *Reason 1*, *Reason 2*, etc. are *question type*s that have been further subdivided. On average, one question sentence includes 1.74 *question keyword*s.

In bidirectional LSTM, word embeddings were randomly initialized and their size was 250. The dropout rate was set to 0.5 and applied to the non-recurrent connections during models training. These hyper parameters were decided by conducting a grid search. We used the Adam optimizer method for model optimization. Parameters of the Adam are (α = 0.0003, β_1 = 0.9, β_2 = 0.999, ϵ = 10^{-8}).

B. Results of Question Type Estimation

In *question type* estimation, the test data was given to the trained models and evaluated in terms of *estimation rates* which counted the number of the correct answers from the output of the models and recall[1] and precision[2]. In addition to the *attention-based bidirectional LSTM* model described in II-C, for comparison, we conducted experiments with a *non-attention unidirectional LSTM* model, an *attention-based unidirectional LSTM* model, the rule-based model constructed in our previous study, SVM and Random Forest.

Table III shows the experimental results. The *attention-based unidirectional LSTM* showed an estimation rate of 90.5% and outperformed the results of the rule-based model, SVM, Random Forest, and the *non-attention undirectional LSTM*. In the case of bidirectional LSTM models, the estimation rate 94.3% and 93.3% were obtained respectively with and without *attention*.

Improvement of the estimation rate by introducing *attention mechanism* was not expected. Since *question type* estimation is a simple estimation problem, we found that a sufficiently high estimation rate can be obtained with only bidrectional LSTM, and the *attention mechanism* is unnecessary.

C. Results of Question Keyword Class Estimation

In *question keyword class* estimation, we evaluated the models in terms of estimation rates which counted the correct answers included in outputs of the model for the test data. In addition to our proposed model (the *attention-based bidirectional LSTM encoder-decoder* model, which uses the outputs from the hidden layers at one and two previous time steps as the input to the decoder, proposed in II-D), we conducted experiments with the conventional *non-attention LSTM encoder-decoder* model [7,8] and the *attention-based LSTM encoder-decoder* model [7], which uses the outputs

[1]the fraction of data that is actually positive among the data predicted to be positive.

[2]the fraction of data that is predicted to be positive among the data is actually positive

TABLE I
DISTRIBUTION OF *QUESTION TYPE* IN THE DATA SET.

Reason1	Reason2	Location1	Location2	Location3
100	62	52	71	56
Definition	Result	Check1	Check2	Check3
87	67	62	54	61
Propose	Select	Common-sence	Strategy	Chat
86	54	59	59	70

TABLE II
DISTRIBUTION OF *QUESTION KEYWORD CLASS* IN THE DATA SET.

Coodination1	Coodination2	Adject-good	Adject-bad	Player
204	197	236	129	92
Denial	Gain-of-stone	Win-or-lose	Move	Term
63	52	36	196	250
Elvaluate	Number-of-stone	Compare	Directive	
87	51	57	85	

TABLE III
EXPERIMENTAL RESULTS OF *QUESTION TYPE* ESTIMATION.

Models	Estimation Rate(%)	Recall	Precision
Rule-based	75.8	-	-
SVM	84.4	0.84	0.85
Random Forest	75.6	0.76	0.80
Non-attention unidirectional LSTM	86.7	0.84	0.83
Attention-based unidirectional LSTM	90.5	0.95	0.84
Non-attention bidirectional LSTM	93.3	0.97	0.86
Attention-based bidirectional LSTM	94.3	0.98	0.86

TABLE IV
EXPERIMENTAL RESULTS OF *QUESTION KEYWORD CLASS* ESTIMATION.

Models	Estimation Rate(%)
Non-attention LSTM Encoder-Decoder[4,7] (one input)	54.3
Attention-based LSTM Encoder-Decoder[7] (attention, two inputs)	57.1
Our proposed model	59.0

from the hidden layer at only one previous time step), and compared their results.

The experimental results are shown in Table IV. Our proposed model achieved a higher estimation rate than the other models, so it can be said that the *attention mechanism* and our proposed method are effective. Investigating *attention* vectors, they tend to weight the beginning of the input sentence. In the conventional LSTM, there was a problem that it was difficult to reflect the information of the first half of the input series, but we think that it could be solved by adding a large weight to the beginning of the input sentence.

D. Result of Joint Model Estimation

The experimental results with the *joint model* are shown in Table V. As mentioned in II-E, the *question type* estimation model in II-C and the *question keyword class* estimation model in II-D were combined. Using joint learning, the estimation rate of *question type* decreased by 3.8% compared to the result of III-B, but the estimation rate of *question keyword class* improved by 2.9% compared with the result of III-C. The reason for the decline of the *question type* estimation rate is that the *question keyword class* estimation rate was originally low, so it influenced the training of the *question type* estimation.

IV. CONCLUSION

In this paper, we proposed the models to estimate *question type* and *question keyword class* in order to estimate the intent of a user's question. In *question type* estimation, the unidirectional LSTM showed a higher estimation rate with the *attention mechanism* than without the *attention mechanism*. Our proposed method, a bidirectional LSTM with the *attention mechanism*, showed the highest estimation rate. In *question keyword class* estimation, the improvement was obtained by using the *attention mechanism* and the outputs from the hidden layers at one and two previous time steps as the input to the decoder. In *joint model* estimation, the estimation rate of *question type* decreased, but the estimation rate of *question keyword class* improved.

TABLE V
EXPERIMENTAL RESULTS OF JOINT MODEL ESTIMATION.

Estimated contents	Estimation Rate(%)	Recall	Precision
question type	90.5	0.91	0.80
question keyword class	61.9	-	-

In future work, regarding joint model, we are trying to devise the way of joint. Also, we calculate reliabilities of *question type* estimation model and *question keyword class* estimation model and investigate how the difference of reliabilities affects joint learning. In addition, since the training data is only 1000 sentences, it seems that models have not been sufficiently trained, so we plan to increase the training data.

ACKNOWLEDGMENT

This work was supported in part by JSPS KAKENHI (Grant No. JP17K00236 and No. JP17H01995).

REFERENCES

[1] P. Haffner, G. Tur, and J. H. Wright, "Optimizing svms for complex call classification," in *Acoustics,Speech,and Signal Processing, 2003, Proceedings.(ICASSP'03). 2003 IEEE International Conference on,* vol. 1, IEEE, 2003, pp. 529-551.

[2] R. Sarikaya, G. E. Hinton, and B. Ramadhadran, "Deep belief nets for natural language call-routing," in *Acoustics,Speech,and Signal Processing (ICASSP), 2011 IEEE International Conference on,* IEEE, 2011, pp. 5680-5683.

[3] G. Mesnil et al., "Using recurrent neural networks for slot filling in spoken language understanding," *Audio,Speech,and Language Processing,IEEE/ ACM Transactions on,* vol. 23, no. 3, pp. 530-539, 2015.

[4] N. T. Vu, "Sequential Convolutional Neural Networks for Slot Filling in Spoken Language Understanding," in *INTERSPEECH,* 2016, pp.3250-3254.

[5] S. Zhu and K. Yu, "Encoder-decoder with focus-mechanism for sequence labelling based spoken language understanding," in *Acoustics, Speech,and Signal Processing (ICASSP), 2017 IEEE International Conference on,* IEEE, 2017, pp. 5675-5679.

[6] T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing,* pp. 1412-1421, Association for Computational Linguistics.

[7] B. Liu and I. Lane, "Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling," in *INTERSPEECH.* 2016, pp. 685-689.

[8] T. Hori et al., "Dialog State Tracking with Attention-Based Sequence-to-Sequence Learning," in *Proc. IEEE Workshop on Spoken Language Technology (SLT).*

[9] MeCab: Yet Another Part-of-Speech and Morphological Analyzer, http://taku910.github.io/mecab/

[10] D. Bahdanau, K. Cho and Y. Bengio, "Neural Machine translation by jointly learning to align and translate," *CoRR,* abs/1409.0473.