


S²-Net: Machine reading comprehension with SRU-based self-matching networks

Cheoneum Park¹  | Changki Lee¹ | Lynn Hong² | Yigyu Hwang³ | Taejoon Yoo³ | Jaeyong Jang⁴ | Yunki Hong⁵ | Kyung-Hoon Bae⁴ | Hyun-Ki Kim⁶

¹Department of Computer Science, Kangwon National University, Chuncheon, Rep. of Korea

²SKtelecom, Seoul, Rep. of Korea

³MindsLab, Seoul, Rep. of Korea

⁴LG Uplus, Seoul, Rep. of Korea

⁵Naver, Seongnam, Rep. of Korea

⁶SW and Contents Research Laboratory, Electronics and Telecommunications Research Institute, Daejeon, Rep. of Korea

Correspondence

Changki Lee, Kangwon National University, Chuncheon, Rep. of Korea.
Email: leeck@kangwon.ac.kr

Funding information

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grants funded by the Korean government (MSIT) (2018-0-00605, Artificial Intelligence Contact Center Solution; 2013-0-00131, Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services).

Machine reading comprehension is the task of understanding a given context and finding the correct response in that context. A simple recurrent unit (SRU) is a model that solves the vanishing gradient problem in a recurrent neural network (RNN) using a neural gate, such as a gated recurrent unit (GRU) and long short-term memory (LSTM); moreover, it removes the previous hidden state from the input gate to improve the speed compared to GRU and LSTM. A self-matching network, used in R-Net, can have a similar effect to coreference resolution because the self-matching network can obtain context information of a similar meaning by calculating the attention weight for its own RNN sequence. In this paper, we construct a dataset for Korean machine reading comprehension and propose an S²-Net model that adds a self-matching layer to an encoder RNN using multilayer SRU. The experimental results show that the proposed S²-Net model has performance of single 68.82% EM and 81.25% F1, and ensemble 70.81% EM, 82.48% F1 in the Korean machine reading comprehension test dataset, and has single 71.30% EM and 80.37% F1 and ensemble 73.29% EM and 81.54% F1 performance in the SQuAD dev dataset.

KEYWORDS

machine reading comprehension, question answering, simple recurrent unit, self-matching network, Korean machine reading comprehension, S²-Net

1 | INTRODUCTION

Machine reading comprehension refers to a machine's ability to understand a given context and determine the correct answer to a question in the context of question-answering applications. For example, machine reading comprehension should be able to understand the context “2004년 건조기 시장에... 의류 건조기 중 LG전자는 점유율 77.4%로 1위를 차지했다. (Machine dryer market in 2004 ... LG Electronics ranked first with 77.4% of clothes dryers.)” and determine

the correct answer output “LG 전자 (LG Electronics)” in the context of a question like “국내 건조기 시장 점유율 1위 누구야? (Who is number 1 in the Korean dryer market share?)”.

Machine reading comprehension has been performed on datasets, such as SQuAD of Stanford, bAbi of Facebook, and MS-MARCO of Microsoft [1–3]. Deep-learning models, such as DrQA, fastQA, R-Net, AoA reader, bidirectional flow (BiDAF), and Match-LSTM [4–9], are currently the focus of research. These deep-learning models perform encoding

and matching of a given context and question, and they use a pointer network model [10] based on an attention mechanism [11] to generate boundary indices (start and end positions) of the correct answer.

The self-matching network used in R-Net has a similar effect to coreference resolution because the self-matching network can obtain contextual information of a similar meaning by calculating the attention weight for its own recurrent neural network (RNN) sequence.

A simple recurrent unit (SRU) is a model that solves the vanishing gradient problem in RNNs by using a neural gate, such as a gated recurrent unit (GRU) [12] or long short-term memory (LSTM) [13]. SRU simplifies the calculation process of memory cells compared to GRU and LSTM by eliminating previous hidden states from the gate input; moreover, it performs parallelization and CUDA-level optimization to show a similar speed to that of a convolutional neural network (CNN), and is 5–10 times faster than LSTM optimized for the CUDA deep neural network library. Moreover, SRU includes a highway network to improve the performance when stacking multilayers [14].

In this paper, we construct a dataset for Korean machine reading comprehension and propose an SRU-based self-matching network (S²-Net) to the encoder RNN using multilayer SRU. To summarize, the main contributions of our model are

1. *Model combination and performance enhancement:* Our model is based on DrQA and fastQA, and it performs additional feature extraction on the question input sequence. We model hidden states by adding alignment weights between words with similar meanings by applying a self-matching network. Our model shows better performance than DrQA for the Korean dataset and SQuAD dataset in Chapter 6.
2. *SRU application for speed improvement:* The deep-learning model requires more computation as the depth of the network increases. We apply SRU to improve the training and testing operation speed of S²-Net, a more complex model than DrQA or fastQA.
3. *Korean dataset construction and model application:* In this paper, we show Korean in the SQuAD style, such as the MindsMRC dataset. We also show the performance for the Korean and SQuAD datasets using S²-Net.

2 | SIMPLE RECURRENT UNIT

SRU is a new recurrent unit model that solves the vanishing (or exploding) gradient problem caused by backpropagation of RNNs with a neural gate, such as GRU and LSTM, and improves the speed by removing the previous hidden state from the gate input [14]. SRU uses an input gate i_t , forget gate f_t , reset gate r_t , and highway network. The equation is as follows.

$$\tilde{x}_t = Wx_t, \quad (1)$$

$$i_t = (1 - f_t), \quad (2)$$

$$f_t = \sigma(W_f x_t + b_f), \quad (3)$$

$$r_t = \sigma(W_r x_t + b_r), \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{x}_t, \quad (5)$$

$$h_t = r_t \odot g(c_t) + (1 - r_t) \odot x_t. \quad (6)$$

The input gate i_t determines whether to reflect the input information by performing element-wise multiplication with \tilde{x}_t , and the forget gate f_t determines whether to reflect the previous internal state information by performing element-wise multiplication with c_{t-1} . \tilde{x}_t is the result of linearly transforming input x_t by weight W_f , where i_t is $i_t = 1 - f_t$ given by (2) and f_t is the result of performing a feed-forward neural network (FFNN) on input x_t and applying the sigmoid function. The forget gate of the existing RNN models (GRU, LSTM) includes the previous hidden state Rh_{t-1} as $f_t = \sigma(W_f x_t + Rh_{t-1} + b_f)$. In contrast, SRU applies FFNN to the gate calculation to reduce the amount of computation and enable parallel computation. c_t is an internal state that controls the information transfer from the input x_t and the previous internal state c_{t-1} , and applies the activation function $g(\cdot)$ to produce the output of the internal state. The hidden state h_t is the result of executing the highway network for the internal state output and the input x_t . In this case, the internal state output $g(c_t)$ determines the extent to which the internal state output is reflected in the hidden state by performing element-wise multiplication with reset gate r_t . The input x_t is calculated using $(1 - r_t)$ and element-wise multiplication to determine whether to reflect the input x_t . Figure 1 shows the SRU.

3 | KOREAN READING COMPREHENSION WITH SRU-BASED SELF-MATCHING NETWORKS (S²-NET)

To perform machine reading comprehension, each model is provided a dataset composed of questions (Q), passages (P), and answers (Y). Each question has m words as $Q = \{q_1, q_2, \dots, q_m\}$, and each passage consists of n words

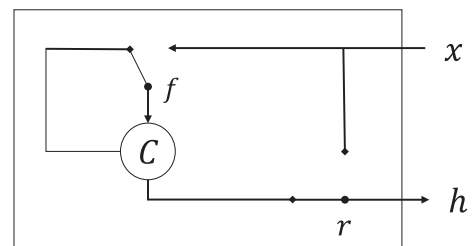


FIGURE 1 Simple recurrent unit

as $P = \{p_1, p_2, \dots, p_n\}$. S²-Net encodes the question and passage to vector representation, and the pointer network outputs a start boundary $y_1(P_{\text{start}})$ and end boundary $y_2(P_{\text{end}})$.

In this paper, we use S²-Net to perform Korean machine reading comprehension, and the S²-Net model is shown in Figure 2. S²-Net performs feature embedding for paragraphs and questions in the feature layer, and performs paragraph encoding and question encoding in the hidden layer. Self-attention is applied to a paragraph encoder vector in the self-matching layer, and the modeling layer models the paragraph encoder vector. Then, output layer points to the answer. The formula for the feature layer is as follows.

$$\tilde{p}_t = [f_{\text{emb}}(p_t); f_{\text{c_emb}}(p_t); f_{\text{exact_match}}(p_t); f_{\text{tf}}(p_t); f_{\text{align}}(p_t)], \quad (7)$$

$$\tilde{q}_t = [f_{\text{emb}}(q_t); f_{\text{c_emb}}(q_t); f_{\text{exact_match}}(q_t); f_{\text{tf}}(q_t); f_{\text{align}}(q_t)], \quad (8)$$

\tilde{p}_t and \tilde{q}_t are the input feature vectors with word embedding, and the extracted features are as follows (\tilde{q}_t is given as the input instead of \tilde{p}_t in the case of a question).

- Word embedding: $f_{\text{emb}}(p_t) = E(p_t)$
- Character embedding: $f_{\text{c_emb}}(p_t) = \text{CNN}(p_t)$
- Exact match: $f_{\text{exact_match}}(p_t) = \mathbb{1}(p_t \in Q)$
- Token feature: $f_{\text{tf}}(p_t) = \text{TF}(p_t)/T$
- Aligned sentence embedding:

$$f_{\text{align}}(p_t) = \sum_i a_{t,i} E(q_i),$$

$$a_{t,i} = \frac{\exp(\alpha(E(p_t)) \cdot \alpha(E(q_i)))}{\sum_j \exp(\alpha(E(p_t)) \cdot \alpha(E(q_j)))}$$

The embedding layer $E(p_t)$ is responsible for mapping each word to its corresponding n -dimensional representation. We used a word-embedding method that uses 2-year news articles of approximately 100,000 words learned from the neural network language model (NNLM) [15]. In this paper, we use a CNN [16] CNN(p_t) for character embedding CNN(p_t), which sets an arbitrary initial value. The character embedding layer maps each word to a high-dimensional vector space. Each character in a word is embedded into vectors using n -size filters, and each output is max-pooled to obtain a fixed-size vector for each word. The exact match feature [4,5] is a binary feature (1 or 0), which verifies that word p_t in the paragraph is included in the question, and each word of the paragraph and the question consists of “morpheme/POS tag”. The token feature normalizes the frequency of each word ($\text{TF}(p_t)/T$, where T is the length of each sequence for the question or passage). The aligned sentence embedding calculates a matching context vector that is multiplied by both the context encoder vector and an alignment score for the paragraph and

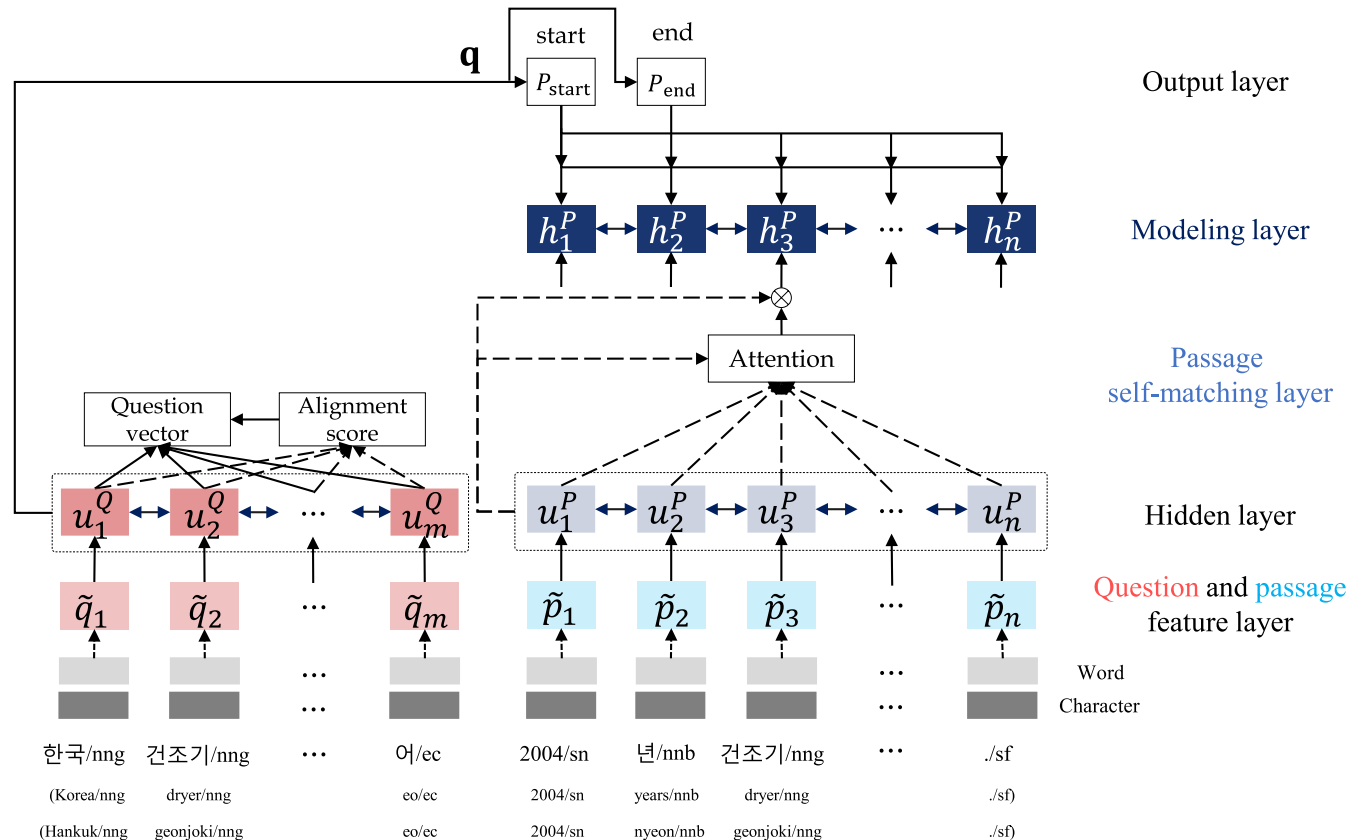


FIGURE 2 SRU-based self-matching network (S²-Net) architecture

question representation. We use a single dense layer $\alpha(\cdot)$ with ReLU nonlinearity. The layer is applied to the output of the word-embedding layer.

Our model makes u_i^Q by encoding the question feature vector \tilde{q}_i . To use u_i^Q in the output layer, we apply attention mechanism on the encoding hidden state u_i^Q . The hidden states of the question sentence to create an alignment vector b_j , and we compute the question vector \mathbf{q} used for both the hidden state of the question sentence and the alignment vector b_j as in the following formula. \mathbf{w} is a weight vector to learn.

$$\mathbf{q} = \sum_j b_j \mathbf{q}_j, \quad (9)$$

$$b_j = \frac{\exp(\mathbf{w} \cdot \mathbf{q}_j)}{\sum_{j'} \exp(\mathbf{w} \cdot \mathbf{q}_{j'})} = \frac{\exp(\mathbf{w} \cdot \mathbf{q}_j)}{\sum_{j'} \exp(\mathbf{w} \cdot \mathbf{q}_{j'})}. \quad (10)$$

The hidden layer that performs paragraph encoding consists of bidirectional SRU (BiSRU), and the formula is as follows, where u_i^P is the hidden state encoded for the paragraph input hidden state \tilde{p}_i .

$$u_i^P = \text{BiSRU}_p(u_{i-1}^P, \tilde{p}_i). \quad (11)$$

u_i^P is used as an input to the modeling layer h_i^P , and is concatenated with the context vector C_i of the self-matching layer, such as $[u_i^P; c_i]^*$, to encode. $[u_i^P; c_i]^*$ is the gated attention-based recurrent network (R-Net), which performs the element-wise product for $[u_i^P; c_i]$ and g_i , which is the nonlinear gate layer with the sigmoid function applied. The formula for the modeling layer h_i^P is as follows.

$$h_i^P = \text{BiSRU}(h_{i-1}^P, [u_i^P; c_i]^*), \quad (12)$$

$$g_i = \text{sigmoid}(W_a [u_i^P; c_i]), \quad (13)$$

$$[u_i^P; c_i]^* = g_i \odot [u_i^P; c_i]. \quad (14)$$

The self-matching layer based on self-matching attention makes context vector c_i from the whole passage for words in passage. The alignment score is calculated by assigning a given sequence as the input (that is, itself) and adjusts the attention weight by calculating a high alignment score between similar hidden states in the input sequence.

$$s_{i,j} = u_i^P W_v^P u_j^P, \quad (15)$$

$$\alpha_{i,i} = \frac{\exp(s_{i,i})}{\sum_{j=1}^n \exp(s_{i,j})}, \quad (16)$$

$$c_i = \sum_{i=1}^n \alpha_{i,i} u_i^P. \quad (17)$$

S²-Net computes the query vector \mathbf{q} and h_i^P , which is created on the modeling layer, as a bilinear sequence attention, and determines the correct answer location for the question in the paragraph. At this time, the output result is the position of the start (P_{start}) and end (P_{end}) of the answer span, \mathbf{W}_s is a weight matrix for the start position, and \mathbf{W}_e is a weight matrix for the end position, as follows.

$$P_{\text{start}}(t) \propto \exp(h_t^P \mathbf{W}_s \mathbf{q}). \quad (18)$$

4 | COMPARISON WITH RELATED NETWORKS

Existing studies to solve machine reading comprehension include the pointer networks based on the attention mechanism, and there are models such as DrQA, fastQA, R-Net, and BiDAF. The proposed S²-Net improves the learning speed and performance for machine reading comprehension by using multiple layers of SRU in the encoder RNN, and we add character (or syllable) representation, some features for the question sentence, and the self-matching layer of R-Net to our model.

4.1 | FastQA

FastQA is simply composed of an embedding layer, encoder RNN, and answer layer. In the embedding layer, word representations and a highway network are applied to the input sequence, and an exact match for each word and aligned sentence representation for question features are extracted and concatenated. In this study, we perform encoding based on SRU in hidden layers. The SRU includes the highway network; thus, there is no need for an additional highway layer, and more layers can be stacked because of the small amount of computation.

The encoder RNN of FastQA uses bidirectional LSTM and shares the weight matrix of the question and paragraph. Thereafter, FFNN is performed, where the weight matrix is applied independently to the question and paragraph vectors. In this study, question and paragraph matching are performed by extracting both the question and paragraph for the aligned sentence representation feature rather than sharing them when encoding the vector of the question and the paragraph. Finally, in the answer layer, FastQA constructs a question vector \mathbf{q} , as in this paper, and it applies the ReLU-based two-layer FFNN with the paragraph-encoding vector to output the beginning and end of the answer. In this study, we calculate the question vector \mathbf{q} and the modeling vector h_i^P as bilinear sequence attention without the ReLU-based two-layer FFNN.

4.2 | DrQA

DrQA consists of a document-retriever module for finding documents related to a question on the web and a document-reader module that performs machine reading comprehension to determine the correct answer to the question from the documents found.

Encoder RNN of DrQA is composed of bidirectional GRU. In this study, we apply the SRU, which has a greater learning speed and higher performance as the layers, including the highway network, are stacked. DrQA does not include the self-matching layer, does not use character representation, and uses the feature vector \tilde{p}_t composed of word representation, exact match, token features, and aligned question representation. Here, the additional token features used are not only $\text{TF}(p_t)$ but also the part-of-speech (POS) information $\text{POS}(p_t)$ and the named-entity information $\text{NER}(p_t)$. In this paper, we did not use any additional features for NER and POS tag information because the input word is a morpheme/POS tag. The token feature is defined below.

$$f_{\text{token}}(p_t) = (\text{POS}(p_t), \text{NER}(p_t), \text{TF}(p_t)). \quad (19)$$

The layer after the paragraph encoding uses the same method in this study (DrQA), with the exception of the self-matching layer. The input of the function to calculate the output result is given by the paragraphs feature embedding p_t instead of the encoding of the modeling layer h_t^p of this study.

$$P_{\text{start}}(t) \propto \exp(p_t \mathbf{W}_s \mathbf{q}), \quad (20)$$

$$P_{\text{end}}(t) \propto \exp(p_t \mathbf{W}_e \mathbf{q}). \quad (21)$$

4.3 | R-Net

R-Net matches the question and paragraph in a gated attention-based matching layer to create a paragraph representation containing the meaning of the question and calculates an alignment score for the matched paragraph representation based on a self-matching attention mechanism. Then, it outputs a result by multiplying the alignment score with an encoded hidden state. In the case of R-Net, word and character representation is used as described in this paper, but instead of the aligned sentence representation, the attention weight is calculated in the question-paragraph matching layer and applied to a hidden state to perform the modeling for the hidden state. R-Net models the question encoding as self-attention and obtains the probability distribution of the attention scores of the output results along with the paragraph encoding when calculating the answer attention weights by pointer networks in the output layer. R-Net applies the concatenation

score to all attention mechanisms, but unlike the attention score method of R-Net, this study uses pointer networks based on bilinear sequence attention to output the position of the answer boundary.

4.4 | BiDAF

BiDAF is a six-layer hierarchical process model based on a bidirectional attention-flow mechanism. The bidirectional attention flow refers to Query2Context $\tilde{\mathbf{H}} = \sum b\mathbf{H} \in \mathbb{R}^{2d \times T}$ and Context2Query $\tilde{\mathbf{U}} = \sum a\mathbf{U} \in \mathbb{R}^{2d \times T}$ [8]. Here, \mathbf{H} is the hidden state from the context word vectors and \mathbf{U} is hidden state from query word vectors. When calculating the alignment score, Query2Context $\tilde{\mathbf{H}}$ and Context2Query $\tilde{\mathbf{U}}$ are calculated together with the paragraph encoding \mathbf{H} using $\beta(\cdot)$ to create the attention weight \mathbf{G} . Then, bidirectional RNN is performed by inputting \mathbf{G} at the modeling layer to produce a new encoding \mathbf{M} .

$$\mathbf{G}_{:t} = \beta(\mathbf{H}_{:t}, \tilde{\mathbf{U}}_{:t}, \tilde{\mathbf{H}}_{:t}), \quad (22)$$

$$\mathbf{M}_{:t} = \text{BiRNN}(\mathbf{G}_{:t}), \quad (23)$$

where $\beta(\cdot)$ is a trainable vector function, such as FFNN after concatenation of inputs [8].

The calculated \mathbf{G} and encoded \mathbf{M} are concatenated to each other at the output layer to output the answer to the question, and the concatenated hidden state is calculated with the linear attention weight to obtain the start (P_{start}) and end (P_{end}) of the answer.

$$P_{\text{start}} = \text{softmax}\left(w_{(P_{\text{start}})}^T [\mathbf{G}; \mathbf{M}]\right), \quad (24)$$

$$P_{\text{end}} = \text{softmax}\left(w_{(P_{\text{end}})}^T [\mathbf{G}; \mathbf{M}]\right). \quad (25)$$

In this study, unlike the bidirectional attention-flow mechanism of BiDAF, we create a paragraph-encoding vector \mathbf{u}_t^p using the aligned sentence representation feature, which is modeled in the self-matching layer to make \mathbf{h}_t^p the input \mathbf{u}_t^p , and bilinear sequence attention is performed in the output layer.

5 | KOREAN MACHINE READING COMPREHENSION DATASET

The dataset of Korean machine reading comprehension consists of paragraphs and question-answer pairs collected from the news and Wikipedia for entertainment and the general domain, and follows a format similar to the SQuAD dataset, as shown in Figure 3.

The dataset (MindsMRC dataset) structure of the Korean machine reading comprehension is based on JSON. In


```

set -
  |- version (str)
  - data[] -
    |- title (str)
    - paragraphs [] -
      |- context_original (str)
      |- dp[] -
        |- head (int)
        |- id (int)
        |- label (str)
        |- weight (double)
        |- text (str)
        - mods [(int)]
      |- qas[] -
        |- question_original (str)
        |- id (int)
        |- question_dp [[(str)]]
        |- question [[(str)]]
        |- answer -
          |- text_original (str)
          |- text [[(str)]]
          |- answer_end (int)
          - answer_start (int)
      - context [[(str)]]

```

FIGURE 3 Korean machine reading comprehension dataset structure

Figure 3, “set” is the key that contains the entire dataset; “data” is the list containing all the data in the current dataset; and “paragraphs” is the list of paragraphs, questions, and answers corresponding to the title of the document. “Context_original” is the raw text collected from the news or Wikipedia, “dp” is a list containing the dependency-parsing results of the text corresponding to context_original, “qas” is a list containing the question and answer information of the document, and “context” is a list containing the morpheme information corresponding to the raw text. The qas is composed of the “question” and “answer” information of the document. It includes the questions, including the morpheme result of the raw question text, and the answer key, including information corresponding to the question. The answer to the question is an answer that consists of answer_start and answer_end, which are the beginning and end index information, respectively, of the answer in the document. An example of the Korean machine reading comprehension dataset is presented in Table 1.

Table 1 lists examples of title, paragraphs, context_original, context, question_original, question, and answer information. The title represents the title of the document (news or Wikipedia), and the paragraphs represent the passage of the document and the question-answer pair. The context_original and question_original are raw texts of the paragraphs and questions, and we produce the results of morphological analysis, such as the context and question, to learn and predict with S²-Net. The answers contain the raw text (text_original) of the answer and the result (text) of the morphological analysis and consist of the start

position (answer_start) and end position (answer_end) of the answer text in the paragraph.

6 | EXPERIMENTAL RESULT

The proposed S²-Net was implemented in PyTorch, and the experiments was performed on a computer with Intel i7-4790 CPU (3.60 GHz), 32-GB RAM, TITAN X (Pascal), and Ubuntu 16.04 OS.

The dataset used in the experiment consisted of 55,088 documents and 139,583 questions for the news domain, and 7,119 documents and 17,948 questions for the Wikipedia domain. The training, development (dev), and test sets were divided as listed in Table 2.

In this study, we conducted the following experiments on Korean machine reading comprehension using S²-Net. Training was conducted using Adam [17], and the learning rate was set at 0.1. The activation functions for the hidden layers and the attention layer are all tanh, and all the RNN layers used the SRU (CUDA-level optimization). Dropout was fixed at 0.2, the number of dimensions of character embedding was 50, the number of dimensions of word embedding was 100, and the number of dimensions in the hidden layer was 128. The CNN for character embedding used window-size (2, 3, 4, 5, 6) filters, and the size of the filter was set to 30. The size of the minibatch was set to 32, and performance evaluation was performed with the dev set for each epoch. We used the performance measurements of exact match (EM) and F1 [1].

In this study, we implemented BiDAF and BiDAF+self-matching (BiDAF+SM) with PyTorch and applied DrQA to the Korean dataset. The performance of each RNN type for BiDAF and BiDAF+SM is listed in Table 3, and the number of layer stacks is (3, 1) (hidden layer, modeling layer). At this time, BiDAF and BiDAF+SM do not use features.

The experimental results showed that the performance of BiDAF is the lowest, at 68.82% F1, and the performance is improved in the order of LSTM and SRU. The SRU was 69.52% F1 (56.46% EM) and exhibited the best performance among the RNN types. In the case of BiDAF+SM, the performance was improved in the order of GRU, SRU, and LSTM. When we used LSTM, the BiDAF+SM showed 68.74% F1, which shows good performance among the RNN types. The BiDAF+SM model exhibited a performance of 54.73% EM using LSTM (RNN type).

Table 4 lists the performance of RNN types for DrQA [4,14], BiDAF, and BiDAF+SM. In all three models, the features ($f_{\text{exact_match}}(p)$, $f_{\text{tf}}(p)$, $f_{\text{aligned}}(p)$) used in DrQA, were added. DrQA did not use character embedding on CNN, and the hidden layer and modeling layer stacks are presented in Table 4.

The experimental results showed that the performance of DrQA improved in the order of GRU, LSTM, and SRU, and

TABLE 1 Example of Korean machine reading comprehension dataset

Title									
장마철에도 뽀뽀하게... 물 만난 의류건조기 Stiff in the rainy season ... well-sold clothes dryers									
Paragraphs									
Context_original	<p>2004년 건조기 시장에 가장 먼저 뛰어든 LG전자를 비롯해 올해 초 삼성전자와 중견 기업까지 건조기 판매에 나서면서 국내 건조기 생산량은 급격히 늘고 있다. 건조기의 대당 판매가격을 고려했을 때 1~2년 내에 연간 시장 규모는 1조 원을 넘을 것으로 예상된다. 국내 건조기 시장은 LG전자가 주도하고 있다. 가격비교사이트 다나와리서치에 따르면 올 1월부터 6월까지 판매된 의류 건조기 중 LG전자는 점유율 77.4%로 1위를 차지했다. 가스식·전기식을 모두 판매하는 LG전자는 올해 초부터 전기식 건조기 사업에 주력하고 있다. 회사는 올해 용량과 사용 편의성을 업그레이드한 트롬 전기식 건조기 신제품 2종을 출시했다. 올해 제품에는 냉매를 순환시켜 발생하는 열을 활용하는 ‘인버터 히트펌프’ 기술을 적용했다.</p> <p>LG Electronics, which was the first to enter the dryer market in 2004, started to sell dryers to Samsung Electronics and mid-sized companies earlier this year. Considering the selling price of dryers, the annual market size is expected to exceed KRW 1 trillion within 1-2 years. The domestic dryer market is dominated by LG Electronics. According to Danawa Research, LG Electronics ranked first, with a market share of 77.4%, among clothes dryers sold from January to June this year. LG Electronics, which sells both gas and electric products, has focused on the electric dryer business from the beginning of this year. The company has released two new TROMM electric dryers that have upgraded capacity and usability this year. This year's products use inverter heat-pump technology that utilizes the heat generated by circulating the refrigerant.</p>								
Context	[[‘2004/sn’, ‘년/nmb’, [‘건조기/nng’], [‘시장/nng’, ‘에/jkb’], [‘가장/mag’], [‘먼저/mag’], [‘뛰어들/vv’, ‘ㄴ/etm’], [‘LG/sl’, ‘전자/nng’, ‘를/jko’], [‘비롯하/vv’, ‘어/ec’], [‘올해/nng’], [‘초/nmb’], [‘삼성전자/nng’, ‘와/jc’], [‘중견/nng’], [‘기업/nng’, ‘까지/jx’], [‘건조기/nng’], [‘판매/nng’, ‘에/jkb’], [‘나서/vv’, ‘면서/ec’], [‘국내/nng’], [‘건조기/nng’], [‘생산량/nng’, ‘은/jx’], [‘급격히/mag’], [‘늘/vv’, ‘고/ec’], [‘있/vx’, ‘다/ef’, ‘/sf’],...]]								
Question_original	<p>한국 건조기 시장 점유율 1위 어딘지 알려줘 Who is in the first place in the Korean dryer market?</p>								
Question	[[‘한국/nng’], [‘건조기/nng’], [‘시장/nng’], [‘점유율/nng’], [‘1/sn’, ‘위/nmb’], [‘어디/np’, ‘이/vcp’, ‘ㄴ 지/ec’], [‘알려주/vv’, ‘어/ec’]]								
Answers	<table border="0"> <tr> <td>text_original</td> <td>LG전자</td> </tr> <tr> <td>text</td> <td>LG Electronics</td> </tr> <tr> <td>answer_start</td> <td>95</td> </tr> <tr> <td>answer_end</td> <td>97</td> </tr> </table>	text_original	LG전자	text	LG Electronics	answer_start	95	answer_end	97
text_original	LG전자								
text	LG Electronics								
answer_start	95								
answer_end	97								

TABLE 2 Count of dataset

	Training set	Dev set	Test set
Number of paragraphs	55,986	6,221	6,220
Number of questions	141,424	8,054	8,054

when we used the SRU, the DrQA model showed 77.04% F1, which is the best performance among the RNN types. BiDAF and BiDAF+SM showed the same pattern as the experimental results in Table 3. The BiDAF model, which was added to features of DrQA, had 78.89% F1, which is an improvement

of 9.37% over the result of BiDAF in Table 3. Second, the performance of BiDAF+SM was 78.38% F1. In the case of EM performance, the BiDAF+SM model was 66.31% better than the others.

Table 5 lists the optimized results of the stacking layer for the DrQA, BiDAF, and BiDAF+SM presented in Table 4. The RNN type used was SRU. We performed optimization using the features of DrQA (DrQA: $f_{\text{exact_match}}(p)$, $f_{\text{tf}}(p)$, $f_{\text{aligned}}(p)$) and the features of our model (Ours: $f_{\text{exact_match}}(p)$, $f_{\text{tf}}(p)$, $f_{\text{aligned}}(p)$, $f_{\text{exact_match}}(q)$, $f_{\text{tf}}(q)$, $f_{\text{aligned}}(p)$, character CNN).

The experimental results showed that the optimal performance of the DrQA model, which was better than other

Model	Layer stack	Modeling layer stack	RNN type	EM	F1
BiDAF	3	1	GRU	55.60	68.82
			LSTM	56.61	69.15
			SRU	56.46	69.52
BiDAF +SM	3	1	GRU	53.33	66.43
			LSTM	54.73	68.63
			SRU	54.47	68.74

TABLE 3 Performance for each RNN type of BiDAF and BiDAF+SM (dev, %)

Model	Layer stack	Modeling layer stack	RNN type	EM	F1
DrQA	3	1	GRU	62.58	75.64
			LSTM	63.58	76.00
			SRU	64.22	77.04
BiDAF	3	1	GRU	64.87	77.45
			LSTM	65.69	77.64
			SRU	66.00	78.89
BiDAF +SM	3	1	GRU	65.42	77.72
			LSTM	66.31	78.38
			SRU	64.53	77.81

TABLE 4 Performance for each RNN type of DrQA, BiDAF, and BiDAF+SM (using features of DrQA, dev, %)

models, was 67.38% EM and 89.90% F1 when we used the features proposed in this paper, and the stack layer was [5, 1]. BiDAF exhibited the best performance of 66.00% EM and 78.89% F1 when using the stack layer [3, 1] and the features of DrQA, and BiDAF+SM also showed 64.53% EM and 77.81% F1 when using the method of BiDAF.

Table 6 presents the performance of the proposed S^2 -Net by RNN type. The stack layer was [3, 1] and we used all the features proposed in this paper.

S^2 -Net exhibited better performance than the other RNN types with 68.33% EM and 80.87% F1 when the RNN type was SRN. The processing speed was 177 document/s, which is approximately 3.4 and 2.5 times faster than LSTM and GRU, respectively.

Table 7 lists the optimal performance of S^2 -Net proposed in this paper, namely, DrQA, BiDAF, and BiDAF+SM. DrQA is defined as the baseline of the experiment. The number of layer stacks was [3, 1], and we used SRU as the RNN type and the features of DrQA for the baseline. We adopted the layer stack number, RNN type, and features with the best performance from Tables 4–6 for each model, and all other hyper-parameters were applied in the same way.

The experimental results showed 68.52% EM and 81.15% F1, which is 4.11%, 2.26%, 2.77%, and 1.25% higher than the F1 of baseline DrQA, BiDAF, BiDAF+SM, and DrQA+BiSRU, respectively, when we used a five-stack hidden layer and two-stack modeling layer, applying SRU

Model	Layer stack	Modeling layer stack	Features	EM	F1		
DrQA	3	1	DrQA	64.22	77.04		
			5	2	DrQA	64.99	77.98
			Ours	67.38	79.90		
BiDAF	3	1	DrQA	66.00	78.89		
			5	2	DrQA	65.94	78.78
			Ours	65.94	78.54		
BiDAF +SM	3	1	DrQA	64.53	77.81		
			5	2	DrQA	57.76	71.72
			Ours	63.71	76.68		

TABLE 5 Optimization of layer stack and features for DrQA, BiDAF, and BiDAF+SM (dev, %)

TABLE 6 Performance for each RNN type of S²-Net (dev, %)

Model	Layer stack [hidden, modeling]	RNN type	EM	F1	document/s
S ² -Net	[3, 1]	GRU	65.74	78.01	70
		LSTM	63.29	75.73	51
		SRU	68.33	80.87	177

and features, as is proposed in this paper. Thus, our model processed 168 sentences/s. At this time, when the ensemble method was applied to S²-Net, we achieved a performance of 70.16% EM and 81.87% F1.

Table 8 presents the performance of the applied S²-Net model ([5, 2], SRU) proposed in this paper compared to the best performance of the test set in Table 7.

The experimental results show that the proposed S²-Net model has the performance of (single) 68.82% EM and 81.25% F1, and (ensemble) 70.81% EM and 82.48% F1 in the test set.

As listed in Table 9, all the features contribute to the performance of our final system S²-Net.

Table 9 shows that without the aligned sentence-embedding feature for the question as $f_{\text{aligned}}(p)$ (both word embedding and character embedding, aligned sentence feature for the passage, and a few manual features), our system still showed 81.13% F1. When we performed our model without exact matching and term frequency for question ($f_{\text{exact_match}}(q) + f_{\text{tf}}(q)$), our system showed 81.04% F1. When the additional gate used in the self-matching layer was removed, the performance was reduced by 0.28%, to 80.87% F1. Without the exact match and term frequency for passage ($f_{\text{exact_match}}(p) + f_{\text{tf}}(p)$), we had 78.74% F1, which is a decrease of approximately 2.41%. In the case of removing the aligned sentence-embedding feature for the passage $f_{\text{aligned}}(q)$,

the performance of our system was reduced by 2.69% more than it was by removing $f_{\text{exact_match}}(p) + f_{\text{tf}}(p)$. We showed that the 77.47% F1 was reduced by 3.68% compared to our model used for all features when we did not use character embedding on CNN. When we removed all the features, the performance dropped drastically.

We evaluated the performance of the S²-Net proposed in this paper on the SQuAD dataset for single. We compared our model with the other competitive models on the SQuAD leaderboard in Table 10. The hyper-parameters of Table 10 for SQuAD dataset are the same as those applied to the Korean experiments except for the number of hidden layer dimensions, 150. Furthermore, the word-embedding dimension was set to 300 using GloVe word embedding [21]. We added the POS and named-entity recognition (NER) tag feature.

In the SQuAD dataset for development, S²-Net (our model) exhibited a best performance with 80.8% F1, and the second-best performance with 71.6% EM. Our model in the SQuAD dataset for ensemble showed performance of 81.5% F1 and 73.3% EM. Moreover, we performed performance measurements by combining S²-Net and ELMo [22], which had performance of 83.1% F1 (74.6% EM) and 2.3% more improvement than the S²-Net model. ELMo is a function of

TABLE 7 Performance of Korean machine reading comprehension (dev, %)

Model	Layer stack	Modeling layer stack	RNN type	Features	Single		Ensemble	
					EM	F1	EM	F1
DrQA (baseline)	3	1	SRU	DrQA (no character)	64.22	77.04	-	-
BiDAF	3	1	SRU	DrQA	66.00	78.89	-	-
BiDAF+SM	3	1	LSTM	DrQA	66.31	78.38	-	-
S ² -Net	3	1	SRU	Ours	68.33	80.87	-	-
DrQA+BiSRU	5	2	SRU	Ours	67.38	79.90	-	-
S ² -Net	5	2	SRU	Ours	68.95	81.15	70.16	81.87

TABLE 8 Performance for S²-Net (test, %)

Model	Layer stack [hidden, modeling]	RNN type	EM	F1
S ² -Net (single)	[5, 2]	SRU	68.82	81.25
S ² -Net (ensemble)			70.81	82.48

TABLE 9 S²-Net ablation (dev, %)

Feature	F1	Δ
Our model (ensemble)	81.87	+0.72
Our model (single)	81.15	–
$f_{\text{aligned}}(p)$	81.13	–0.02
$f_{\text{exact_match}}(q) + f_{\text{if}}(q)$	81.04	–0.11
Additional gate	80.87	–0.28
$f_{\text{exact_match}}(p) + f_{\text{if}}(p)$	78.74	–2.41
$f_{\text{aligned}}(q)$	78.46	–2.69
Character CNN	77.47	–3.68
No_features	70.98	–10.17

TABLE 10 Performance of single S²-Net model with other competitors on the SQuAD (dev, single, %)

Model	EM	F1
Match-LSTM with Bi-Ans-Ptr [9]	64.1	73.9
RaSoR [18]	66.4	74.9
Multi-perspective matching [19]	66.1	75.8
BiDAF [8]	67.7	77.3
DrQA [4]	69.5	78.8
Smarnet [20]	71.4	80.2
r-net [6]	72.3	80.6
S ² -Net (ours)	71.6	80.8
S ² -Net (ensemble)	73.3	81.5
S ² -Net with ELMo	74.6	83.1
Human performance [1] (test)	82.3	91.2

the internal states of a deep bidirectional language model (biLM) that is pretrained on a large text corpus.

Finally, we show the training time of S²-Net and other models in Table 11. BiDAF and R-Net are our implementations, and all models were tested on PyTorch. An RNN-type S²-Net was SRU, the others were GRU, and all other hyper-parameters for all models were the same. As a result, DrQA was the fastest training model in both the SQuAD and MindsMRC datasets, and S²-Net was the second fastest model. S²-Net showed a slow training speed of 151 s with DrQA in the SQuAD dataset, but a faster learning speed than BiDAF or R-Net by over 678 s. In the MindsMRC dataset, S²-Net showed a training speed that was 37 s slower than that of DrQA, but faster than BiDAF or R-Net by 1,047 s. Our model showed a better F1 score than R-Net and a training speed approximately 2.3–3.1 times faster.

7 | CONCLUSION

In this paper, we proposed a Korean machine reading comprehension model using an SRU-based self-matching

TABLE 11 Training time of S²-Net with other models (dev, seconds per epoch)

	Training time
Model (SQuAD dataset)	
DrQA	191
BiDAF (our implementation)	1,020
R-Net (our implementation)	1,071
S ² -Net (our implementation)	342
Model (MindsMRC dataset)	
DrQA	764
BiDAF (our implementation)	1,848
R-Net (our implementation)	1,898
S ² -Net (our implementation)	801

network (S²-Net), described the Korean machine reading comprehension dataset and the method of constructing the dataset, and performed an experiment comparing S²-Net, DrQA, DrQA+BiSRU, BiDAF, and BiDAF+SM.

The experimental results showed that S²-Net exhibited the best performance with the Korean machine reading comprehension dataset, with (single) 68.95% EM and 81.15% F1, and (ensemble) 70.16% EM and 81.87% F1 when the five-layer stack for the hidden layer and the two-layer stack for the modeling layer and data is the dev set. For the same hyper-parameters, our system achieved performance of (single) 68.82% EM and 81.25% F1, and (ensemble) 70.81% EM and 82.48% F1 in the test set. In comparison, DrQA+BiSRU (stack layer [5, 2], RNN type: SRU, using our features) showed 67.38% EM and 79.90% F1, and BiDAF (stack layer [3, 1]; RNN type: SRU, using DrQA features) showed 66.00% EM, 78.89% F1. BiDAF+SM (stack layer [3, 1]; and RNN type: LSTM, using DrQA features) showed 66.31% EM and 78.38% F1 performance. In addition, our model achieved 71.6% EM and 80.8% F1 for the single model and 73.3% EM and 81.5% F1 for the ensemble model in the SQuAD dev dataset.

Future studies will build further training data for Korean machine reading comprehension and will apply models such as hierarchical RNNs.

ACKNOWLEDGMENTS

This work was supported by Institute for Information & Communications Technology Promotion (IITP) grants funded by the Korean government (MSIT) (2018-0-00605, Artificial Intelligence Contact Center Solution; 2013-0-00131, Development of Knowledge Evolutionary WiseQA Platform Technology for Human Knowledge Augmented Services).

ORCID

Cheoneum Park  <https://orcid.org/0000-0001-5386-0483>

REFERENCES

1. P. Rajpurkar et al., *SQuAD: 100,000+ questions for machine comprehension of text*, arXiv preprint arXiv:1606.05250, 2016.
2. F. Hill et al., *The goldilocks principle: reading children's books with explicit memory representations*, arXiv preprint arXiv:1511.02301, 2015.
3. T. Nguyen et al., *MS MARCO: A human generated machine reading comprehension dataset*, arXiv preprint arXiv:1611.09268, 2016.
4. D. Chen et al., *Reading Wikipedia to answer open-domain questions*, arXiv preprint arXiv:1704.00051, 2017.
5. D. Weissenborn, G. Wiese, and L. Seiffe, *Making neural QA as simple as possible but not simpler*, in Proc. 21st Conf. Comput. Nat. Lang. Learning (CoNLL 2017), Vancouver, Canada, 2017.
6. W. Wang et al., *Gated self-matching networks for reading comprehension and question answering*, in Proc. 55th Annu. Meeting Assoc. Comput. Linguistics, Vancouver, Canada, July 2017, pp. 189–198.
7. Y. Cui et al., *Attention-over-attention neural networks for reading comprehension*, arXiv preprint arXiv:1607.04423, 2016.
8. M. Seo et al., *Bidirectional attention flow for machine comprehension*, arXiv preprint arXiv:1611.01603, 2016.
9. S. Wang and J. Jiang, *Machine comprehension using match-LSTM and answer pointer*, arXiv preprint arXiv:1608.07905, 2016.
10. O. Vinyals, M. Fortunato, and N. Jaitly, *Pointer networks*, in Adv. Neural Inform. Process. Syst., Montreal, Canada, 2015, pp. 2674–2682.
11. D. Bahdanau et al., *Neural machine translation by jointly learning to align and translate*, Proc. ICLR '15, arXiv:1409.0473, 2015.
12. K. Cho et al., *Learning phrase representation using RNN encoder-decoder for statistical machine translation*, in Proc. EMNLP '14, Doha, Qatar, Oct. 25–29, 2014.
13. S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Comput. **9** (1997), no. 8, 1735–1780.
14. T. Lei and Y. Zhang, *Training RNNs as fast as CNNs*, arXiv preprint arXiv:1709.02755, 2017.
15. C. Lee, J. Kim, and J. Kim, *Korean dependency parsing using deep learning*, in Proc. KIISE HCLT, 2014, pp. 87–91 (in Korean).
16. Y. Kim, *Convolutional neural networks for sentence classification*, in Proc. EMNLP '14, Doha, Qatar, Oct. 25–29, 2014.
17. D. Kingma and J. Ba. Adam, *A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, 2014.
18. K. Lee et al., *Learning recurrent span representations for extractive question answering*, arXiv:1611.01436, 2017.
19. Z. Wang et al., *Multi-perspective context matching for machine comprehension*, arXiv:1612.04211, 2016.
20. Z. Chen et al., *Smarnet: Teaching machines to read and comprehend like human*, arXiv:1710.02772, 2017.
21. J. Pennington, R. Socher, and C. Manning, *Glove: Global vectors for word representation*, in Proc. EMNLP '14, Doha, Qatar, Oct. 25–29, 2014, pp. 1532–1543.
22. M. E. Peters et al., *Deep contextualized word representations*, Int. Conf. Learning Representations, 2018. <https://openreview.net/pdf?xml:id=S1p31z-Ab>.

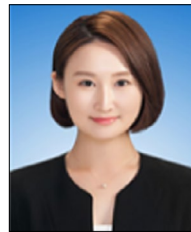
AUTHOR BIOGRAPHIES



Cheoneum Park received his BS and MS degrees in computer science from Kangwon National University, Chuncheon, Rep. of Korea, in 2014 and 2016, respectively. He is now a PhD student at Kangwon National University. His research interests include natural language processing, natural language understanding, question answering, and deep learning.



Changki Lee received his BS degree in computer science from the Korea Advanced Institute of Science and Technology, Daejeon, Rep. of Korea, in 1999. He received his MS and PhD degrees in computer engineering from POSTECH, Pohang, Rep. of Korea, in 2001 and 2004, respectively. From 2004 to 2012, he was a researcher with the Electronics and Technology Research Institute (ETRI), Daejeon, Rep. of Korea. Since 2012, he has been a professor of computer science at Kangwon National University, Chuncheon, Rep. of Korea. His research interests include natural language processing, machine learning, and deep learning.



Lynn Hong received her BS and MS degrees in Library and Information Science from Yonsei University, Seoul, Rep. of Korea, in 2014 and 2017, respectively. She is now working at SKtelecom, Seoul. Her research interests include natural language processing and generation, word sense disambiguation, question answering, and machine learning.



Yigyu Hwang received his BS, MS, and PhD degrees in computer science from Chonbuk National University, Jeonju, Rep. of Korea, in 1993, 1995, and 2001, respectively. From 2001 to 2010, he was a researcher with the Electronics and Technology Research Institute (ETRI), Daejeon, Rep. of Korea. From 2010 to 2012, he was a researcher with Daumsoft. Since 2013, he has been a researcher at MindsLab, Seoul, Rep. of Korea. His research interests include natural language processing.



Taejoon Yoo received his BS degree in computer science from Seoul National University, Seoul, Rep. of Korea, in 1990. He is a CEO at MindsLab.



Jaeyong Jang received his BS degrees in Earth Science Education from Seoul National University in 1996. He works at LG Uplus. He developed the IPTV head-end system and NLP engines in AI. His research interests include natural language processing, deep learning, computer vision, and image recognition.



Yunki Hong received his BS degree in Computer Engineering from Dong-eui University, Rep. of Korea in 2000, and he received his MS degree in computer science from KEIO University, Japan, in 2003. He also completed a doctoral course at KEIO University, Japan, in 2012. He works at NAVER Corporation. He researches the Chatbot engine. His research interests include ontology, question answering, natural language processing, machine learning, and deep learning.



Kyung-Hoon Bae received his MS and PhD degrees in electronic engineering from Kwangwoon University, Seoul, Rep. of Korea, in 2013 and 2016, respectively. He is now vice president of LG Uplus's AI Technology Center. His research interests include natural language processing, visual recognition, machine learning, and deep learning.



Hyun-Ki Kim is a researcher in the language intelligence research group at the Electronics and Telecommunications Research Institute (ETRI), Daejeon, Rep. of Korea. He received the BS and MS degrees in Computer Science from ChunbukNational University, Chonju, Rep. of Korea, in 1994 and 1996, respectively. He received his PhD in computer engineering from the University of Florida, Gainesville, USA, in 2005. His research interests include natural language processing, machine learning, question answering, and social big data analytics.