ICIC Express Letters
Volume 15, Number 7, July 2021

ICIC International ©2021 ISSN 1881-803X
pp. 701–709

ANSWER GENERATION WITH AN ATTENTION MECHANISM DUAL ENCODER LSTM

Andry Chowanda[1], Rhio Sutoyo[1], Meiliana[1] and Sansiri Tanachutiwat[2]

[1]Computer Science Department
School of Computer Science
Bina Nusantara University
JL. K. H. Syahdan No. 9, Kemanggisan, Palmerah, Jakarta 11480, Indonesia
{ achowanda; rsutoyo; meiliana }@binus.edu

[2]Sirindhorn International TGGS
King Mongkut's University of Technology North Bangkok
1518 Pracharat 1 Road, Wongsawang, Bangsue District, Bangkok 10800, Thailand
sansiri.t@tggs.kmutnb.ac.th

Received October 2020; accepted January 2021

Abstract. *Along with the advancement of machine learning, many researchers are trying to build various intelligent applications which are able to mimic humans' capabilities. One of the problems to be solved in the field of question and answering application is answer generation. Such system is able to generate relevant and proper answers for various inquiries asked by the human users. In this research, we explored an idea to solve this challenge by utilizing a modified dual encoder LSTM architecture with an attention mechanism. We proposed a deep learning architecture which consists of eight layers (i.e., two layers for embedding, one layer for attention mechanism, one for thought vector, and four layers for LSTM). The system processes 220,579 English dyadic conversational exchanges between 10,292 pairs of movie characters. There are a total of 304,713 utterances which were trained into the proposed dual encoder LSTM with an attention mechanism using stochastic gradient descent optimizer. As a result, the trained generative conversation model achieved the best training loss of 3.5 in 30 epochs.*

Keywords: Answer generation, Dyadic conversation, Attention mechanism, Question answering, Long short-term memory, Deep learning

1. Introduction.

The development of intelligent applications currently has begun to position computers to become as capable as humans. For instance, they are programmed to communicate as partner in online learning [1], expert consultation [2], customer service application [3], and other services. Ergo, answer and question generation as the main component of dyadic conversation, has become an important research area. Dyadic conversation refers to a conversation engaged between two people, usually in face to face interaction [4]. In this preliminary research, we focus on answer generation problems which are able to respond to various user inquiries appropriately. Unlike humans, computer is not adequately equipped with the skill to understand the natural language of human conversation (e.g., English, and Indonesian). In addition, computer is a social-ignorance conversational agent which will most likely affect the overall conformity of generated answers. Those aforementioned shortcomings underlie our research questions that will be facilitated in this work.

There are two general approaches in answer generation, which are retrieval and generative. The first approach is based on the concept of question-answering method. The system has to be able to provide a complete list related to the selected conversation topic (e.g., movies, sports, or recent news). This approach is easy-to-follow and built, but the

DOI: 10.24507/icicel.15.07.701

static answer provided in the knowledge base could cause a rigid conversation between the agents. Moreover, it requires massive effort in data collection process.

On the other hand, the generative approach is able to generate answers based on a trained statistical language model. This approach promises dynamic responses for the human interlocutors, which enables a lively conversation. However, the accuracy of generated answers is highly dependent on the trained computational model.

This research is interested in the generative approach to generate answers for dyadic conversation by applying multi-layered long short-term memory (LSTM) architecture. Furthermore, an attention mechanism is also applied to learning information that is considered essential and should not be forgotten in the feedback loops of LSTM. The exploration of the modified dual encoder and LSTM architecture to train answer generation model is proposed as the contribution in this research. Our research is conducted by utilizing Cornell Movie-Dialogs Corpus [5] which consists of 3,164 annotated words, 158 utterances, and 79 dialogues. As a result, the conducted experiment shows a promising result from our trained answer generative model. The system achieves the best training loss of 3.5 in 30 epochs.

The paper is organized as follows. Recent works done in this field are thoroughly described in the next section. Next, the proposed method of this research is presented in the Proposed Method section. Furthermore, the results are presented and discussed in the Results and Discussion section. Finally, the last section presents the research implication and future research direction.

2. **Recent Work.**

In this research project, we focus on building a deep-learning architecture for a neural generation natural language system which is able to generate answers/responses by utilizing multi-layered long short-term memory (LSTM) technique [6, 7]. Intelligent & relatable answers should be generated from the system by extracting, scraping, and combining natural language text from the knowledge base.

2.1. **Answer generation.**

Text generation is one of challenging problems in the natural language processing (NLP) research field. There are two types of text generation in a conversation: question and answer generations. The goal of question generation is to generate human-like questions from a variety of inputs, i.e., natural language text [8], knowledge base [9], and picture [10]. With the rise of online learning (e.g., Udemy and Coursera) through Massive Open Online Courses (MOOCs), the question generation task might become an essential role as learning outcome (LO) evaluator [11]. Such technique might be used to create a list of questions based on the given materials (e.g., essay, story, and paragraph). On the other hand, answer generation approach tries to pick, filter, rate, and combine available words, phrases, or part of sentences from the given corpora and/or public knowledge base resources to generate human-like answer [12]. To the best of our knowledge, there are two techniques to generate answers: rule-based [13] and machine learning [14].

Template of responses is created in the rule-based technique as database of predefined answers. They are generally created in the form of XML or JSON format. First, important keywords (e.g., conversation topics, part-of-triplets, and information state) are extracted from the given questions. Then, the information is used to select the most relevant template as the answer. Part-of-triplets (i.e., subject, predicate, and object) are inserted to the selected template to make the answer become relevant. Sutoyo et al. used this technique in their paper to build an emotionally realistic chatbot framework [13].

In their paper, Fu and Feng created an answer generation system with heterogeneous memory [14]. Their method is able to generate results with related information, resulting in enriched generated responses, to answer the given questions. Heterogeneous memory in their system contains two formats, which are knowledge triples and semi-structured entities from information extraction (IE) tasks. The triple, or also known as triplets, is a structured natural language text in the form of subject-predicate-object [15, 16]. Furthermore, the semi-structured entities are generally acquired from the information extraction process and stored in the form of keywords [17]. They modified and utilized WikiMovies[1] datasets for their research purposes. The corpus contains 115 question patterns and 194 answer patterns and covers ten topics. Both of the components (i.e., triples and IE results) are combined in the memory data format as key and value. As a result, the combination approach provides rich knowledge base for its question and answering system. For instance, the generated answer of A_4 in Table 1 is more evident to the users than A_1 because it has more details (i.e., director name and year) related to the given questions.

Table 1. Answer sentences generated by different question-answering systems [14]

Q	**Who is the director of the Titanic?**
A_1	James Cameron
A_2	James Cameron directed the Titanic.
A_3	James Cameron directed it.
A_4	James Cameron directed it in 1999.

2.2. **Cornell Movie-Dialogs Corpus.** We use publicly available conversational datasets shared by previous researches for our corpus resources. In the beginning, we would like to use Indonesian dyadic conversation dataset provided by Tho et al. [18]. The corpus has three default topic categories (i.e., food, travelling, and future), but the participants could also discuss their own topic (e.g., student activities). Each session of conversation is approximately eight to ten minutes. The dataset contains 3,164 annotated words, 158 utterances, and 79 dialogues. Nevertheless, the average length of sentences is lengthy, i.e., 105.2 max words/utterances, and shaped like a monologue. Thus, the conversation corpus is not suitable for our research. Based on further findings, we decided to utilize Cornell Movie-Dialogs Corpus [5]. The dataset contains 220,579 English conversational exchanges between 10,292 pairs of movie characters, resulting 304,713 utterances. Each of utterance has been annotated into question and answer. Example of those pairs from Cornell Movie-Dialogs Corpus is shown in Table 2.

Table 2. An example of question and answering pair from Cornell Movie-Dialogs Corpus [5]

Q_1:	She okay?	A_1:	I hope so.
Q_2:	You realize what is at stake?	A_2:	Our lives, the revolution, my career?
Q_3:	Hey, buddy, that boat still runs, eh?	A_3:	Yeah, it still runs.
Q_4:	When are you going to call them?	A_4:	About what?
Q_5:	Freeze! You're busted!	A_5:	What are you gonna do about it?
Q_6:	Where?	A_6:	The airport. Trying to get on the plane leaving for the States.

2.3. **Multi-layered long short-term memory architecture.** Long short-term memory, or can be shortened as LSTM, is an artificial recurrent neural network architecture which has feedback connections through its three information gates, i.e., input, output, and forget [19]. The input gate functions as a filter for new data which is passed into the cell. Unrelated or useless information is removed by the forget gate in order to handle exploding and vanishing gradient problems. Lastly, the output gate is utilized to result

[1]http://fb.ai/babi

part-of-information, which is relevant with its objectives (e.g., word predictions). In general, each cell of LSTM architecture variants comprises those three gates. However, some variations may have less or more information gates, e.g., gated recurrent units do not have an output gate.

In their paper, Sutskever et al. proposed a multi-layered long short-term memory to improve performance on challenging learning tasks, i.e., machine translations [20]. Their architecture contains two major parts, which are encoder and decoder. Each part might have multiple-layer of LSTM networks. The prior part encodes sequence of inputs into a fixed dimension vector. Moreover, the function of the latter part is to decode results from the processed vector. An example architecture of multi-layered LSTM can be seen in Figure 1.

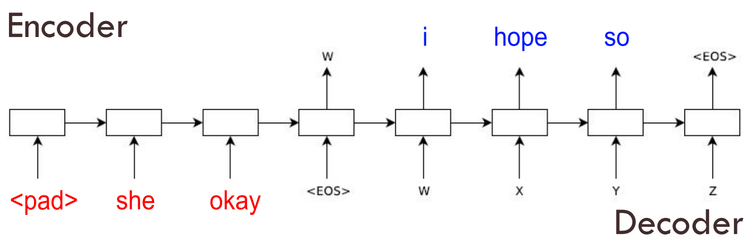

FIGURE 1. Example architecture of dual encoder long short-term memory [20]

In this research, we borrowed the concept of attention mechanism from Bahdanau et al. [21]. We apply multi-layered long short-term memory architecture combined with an attention mechanism to performing answer generation.

Some researches have been done to model the answer generation based on the user's inputs. Liu and Guo [22] proposed a bi-direction LSTM to process text and compare the model against CNN architecture. The results indicate that bi-direction LSTM is superior to CNN architecture. Santhanam [23] proposed an LSTM architecture with a context vector to generate text for a given set of input from the user. The model achieved a training accuracy in a range of 67% and 85%. The effort to train conversation model in other languages compared with English has also been popular. Chowanda and Chowanda [24] proposed trained text vector model in the Indonesian language. The best result achieved from the model was 2.37% unknown words, the loss rate was 1.66, and the perplexity of 4.96. Islam et al. [25] proposed an LSTM network to train a conversation model in Bangla language. However, the performance metrics were not implicitly shown in their paper.

3. **Proposed Method.** Deep learning techniques have been a prominent approach to solve several problems in natural language processing, i.e., conversation system. There are generally two major approaches that can be adapted to solve problems in conversation processing, which are retrieval and generative. Each has its advantages and disadvantages. The retrieval technique ensures that all answers are prepared or hand-crafted beforehand. Hence, all questions should have answers. However, the corpus preparation process demands tremendous resources (e.g., time and people) to prepare all possible combination of query-answer pairs.

On the other hand, the generative technique uses conversation model to produce answers. Hence, the hand-crafted answers are not fully required. However, problems might arise if the trained models are not enough to learn "the grammar" from data. If the data for training are inadequate, the generated sentences might sound like broken sentences. In this research, we are interested in the generative techniques with abundant data for training the conversation models. Figure 2 illustrates the proposed method for the learning

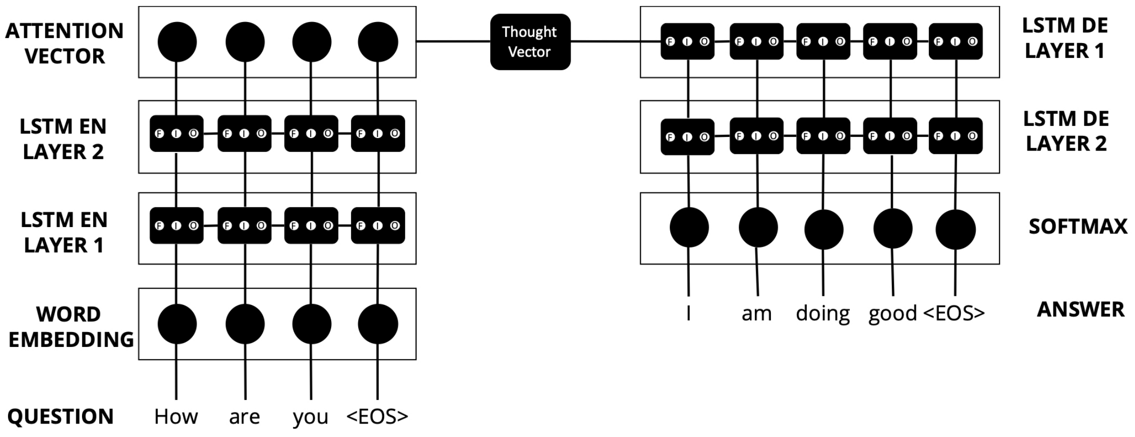

FIGURE 2. Proposed architecture

algorithm using dual encoder long short-term architecture [20] with an attention mechanism [21]. The dual encoder long short-term architecture is the most suitable architecture to learn sequential data such as conversation pattern. Moreover, it is also considered as the best architecture for problems that have different number of inputs and outputs, such as question and answer data.

The proposed method is inspired by the sequence to sequence model [20] combined with the attention mechanism technique [21]. As the first step, the question sentence is transformed into a representable format to the LSTM architecture using Word2Vec word embedding. With Word2Vec, the words in the question sentence are transformed into vector space. The vector is then passed into the first layer of the LSTM encoder layer. The learned state in the first encoder layer is then passed into the second layer of the LSTM encoder layer. Before the state from the second layer is passed to the thought vector, an attention mechanism is implemented to learn which information should be remembered from the data. This is due to the fact that in the LSTM or GRU, there is a good chance that important information is missing in the process through the layers as the data get bigger and bigger [21]. The attention mechanism will learn the weight of the model at time t described in the work done by [21]. In the next process, the thought vector from the encoder layers is passed to the first layer of the LSTM decoder layer. Similar to the encoder layer, the learned state in the first layer is passed into the second LSTM decoder layer. Finally, the Softmax layer determines the best possible answer generated word by word.

In this research, Cornell Movie-Dialogs Corpus [5] is used to train a conversation model in the English language. The corpus was chosen because the number of conversation in the corpus is relatively big and covers quite general topics. Moreover, the questions and answers in the corpus were also annotated. To increase the quality of the data, the corpus was pre-processed before it was trained. The pre-processing techniques and training mechanism, as well as the results, are thoroughly described in the Results and Discussion section.

4. **Results and Discussion.**

Total of 220,579 English conversational exchanges between 10,292 pairs of movie characters, resulting 304,713 utterances were trained into the proposed dual encoder LSTM with an attention mechanism using stochastic gradient descent optimizer. The corpus was trained to generate conversation (i.e., answers) based on the questions asked. Some pre-processing techniques were applied before the corpus admitted into the dual encoder LSTM with an attention mechanism architecture. The pre-processing techniques aim to increase the training performance as well as lower the loss or error [26, 27]. The pre-processing techniques are bucketing and padding. The first pre-processing step done was to create all words length in the sentence same with a fixed length. In order to achieve that, the sentences were reconstructed with padding symbols to fill out the blank caused by the fixed-length problem. There are four padding symbols used: $< BOS >$, $< EOS >$, $< PAD >$, and $< UNK >$. The $< BOS >$ symbol is used to mark the beginning of the sentence in the decoder layers, marking when the architecture starts to generate the words. In contrast, the $< EOS >$ signs the end of the sentence in the encoder layers. The $< PAD >$ is used to fill out the blank space caused by the fixed-length system. Finally, the $< UNK >$ is used when a word is not represented in the vocabulary words vector. Table 3 illustrates the example of padding and bucketing example.

Table 3. Padding and bucketing example

Techniques	**Process & Implementation**
Sentence 1Q	Q_1: She okay?
Sentence 1A	A_1: I hope so.
Padding 1Q	Q_1: [$< PAD >$, $< PAD >$, She, Okay, $< EOS >$]
Padding 1A	A_1: [$< BOS >$, hope, so, $< PAD >$, $< PAD >$]
Bucketing	Sentence 1 belongs to bucket (5,10)
Sentence 2Q	Q_2: You realize what is at stake?
Sentence 2A	A_2: Our lives, the revolution, my career?
Padding 2Q	Q_2: [$< PAD >$, $< PAD >$, $< PAD >$, You, realize, what, is, at, stake, $< EOS >$]
Padding 2A	A_2: [$< BOS >$, $< PAD >$, $< PAD >$, $< PAD >$, Our, lives, the revolution, my, career]
Bucketing 2	Sentence 1 belongs to bucket $(10, 20)$

The next pre-processing applied was the bucketing. Fixed length of words in the sentences leading to the inefficiency of memory and process usage, as the process and memory are usually determined by the longest number of words in all the sentences in the corpus. For example, if the longest sentence in the corpus has 40 words, then we will need 38 padding symbols to a sentence that only has two words. This can be easily solved with bucketing technique by creating a list of buckets for the word vectors. Hence, a sentence with 40 words will not fall into the same bucket with a sentence with only three words. List of bucket implemented in this research is $[(5, 10); (5, 15); (10, 15); (10, 20); (15, 20); (15, 30); (20, 25); (20, 40)]$. Supposed the first sentence (i.e., the interlocutors 1) has six words, and the second sentence (i.e., interlocutor 2) has eight words, then the first sentence is reconstructed with 4 padding symbols and six words, while the second sentence is reconstructed with 2 padding symbols and eight words. Those sentences belong to the bucket system of $(10, 20)$. The first digit represents the length of the first sentences with padding symbols, and the second digit represents the length of both sentences with padding symbols. Other examples of implementation can be seen in Table 3.

The processed corpus was trained with the dual encoder LSTM with an attention mechanism architecture. The architecture consists of eight layers, with two layers for embedding, one layer for attention mechanism, one for thought vector, and four layers for the LSTM layers (one input, one soft-max, and two hidden). The LSTM layers consist of 128 nodes per-layer. The initial hyperparameters tuned to the architecture were 128 sizes of the batch, 100 ephocs, and the initial learning rate (α) was set 2.0 with a decay rate of 0.5 per-epoch. The corpus was trained with a high-performance computer with a 12GB Titan V GPU for more than 143.7 hours. Due to the limited resources (i.e., time and memory), the training, with the size of vocabulary 24,000 vocabularies, was limited to only 30 epochs, instead of 100.

The first training process was to convert the words into a vector space. This process enables the architecture (i.e., the LSTM nodes) to statistically process the words. This allows us to make a correlation between words as well as to draw a pattern from the corpus. The architecture requires two sentences as the training inputs. The first sentence (i.e., the question sentence) will be passed into the LSTM encoding layers, and the second sentence (i.e., the answer sentence) will be passed into the LSTM decoding layers. After the words were converted into a vector representation of words, the question vector was passed to the first layer of LSTM encoding layer for training. The first LSTM encoding layer serves as an input layer, while the second LSTM encoding layer serves as the hidden layer to learn the states from the input layer. Before the learned states are passed into the thought vector, the attention mechanism is implemented. The attention mechanism would help the model to remember the important information from the LSTM encoding layer by adjusting the weight of the information passed (see [21]). The states are then passed into the LSTM decoder layers through the thought vector [20, 28]. In the training phase, the Softmax layer can serve as the vector representation of words for the answer sentence, while, in the generating answer phase, the Softmax layer serves as the probability layers for words to be generated.

Figure 3 illustrates the results of 143.7 hours of the conversation model training in 30 epochs. The average time required for training was 4.8 hours per-epoch. In the first epoch, the average loss achieved was 6.3 and decreasing steadily to the value of training loss of 3.5 in the thirtieth epoch (more than half or 56% from the initial training loss).

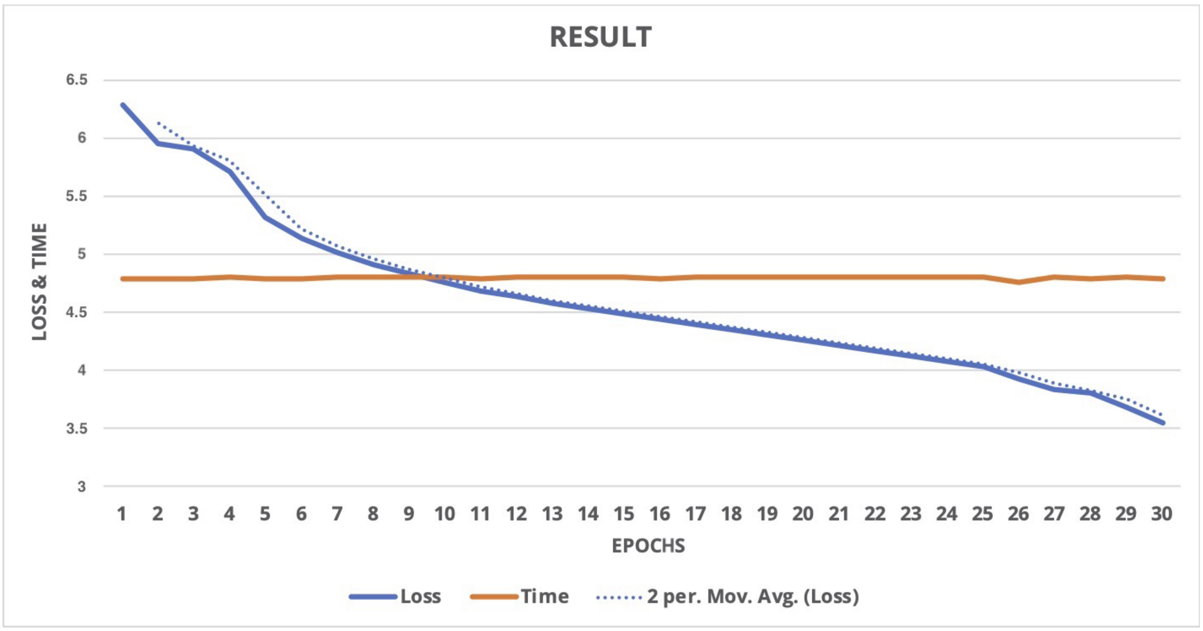

Figure 3. Training result

5. **Conclusion and Future Work.** This work proposes a contribution for the natural language processing and deep learning community by exploring the modified dual encoder LSTM architecture to train generative conversation models with a reasonably huge corpus. More than 143.7 hours of training was done to model generative conversation from 304,713 pairs of utterances using a high computing power computer with a 12GB Titan V GPU. The generative conversation model trained achieved the best training loss of 3.5 in 30 epochs. The trained generative conversation model can also be implemented

to any systems that required natural language processing (i.e., conversation), for example: chat-bot [29, 30] or Embodiment Conversational Agent [28]. For the future research direction, a huge multi-modal conversation dataset in other languages is being collected (e.g., Indonesian and Thailand). Once the collection process is done, the dataset will be trained to the proposed architecture. This will contribute to the generative conversation model in those languages.

Acknowledgment. This work is supported by Directorate General of Research and Development Strengthening, Indonesian Ministry of Research, Technology, and Higher Education, as a part of Penelitian Terapan Unggulan Perguruan Tinggi Research Grant to Binus University titled "Perancangan Model Komputasi untuk Komunikasi Dyadic dengan Menggunakan Algoritma Deep Learning" or "Designing a Computational Model for Dyadic Communication Using a Deep Learning Algorithm" with contract number: 225/SP2H/LT/DRPM/2019, 088/LL3/PG/2020, 039/VR.RTT/IV/2019. We also would like to extend our gratitute to NVIDIA Corporation with the donation of the Titan V Pascal GPU used for this research.

REFERENCES

[1] D. Song, M. Rice and E. Y. Oh, Participation in online courses and interaction with a virtual agent, *International Review of Research in Open and Distributed Learning*, vol.20, no.1, 2019.

[2] L. Ni, C. Lu, N. Liu and J. Liu, MANDY: Towards a smart primary care chatbot application, in *Knowledge and Systems Sciences. KSS 2017. Communications in Computer and Information Science*, J. Chen, T. Theeramunkong, T. Supnithi and X. Tang (eds.), Singapore, Springer, 2017.

[3] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan and M. Zhou, SuperAgent: A customer service chatbot for e-commerce websites, *Proc. of ACL 2017, System Demonstrations*, pp.97-102, 2017.

[4] P. Linell, L. Gustavsson and P. Juvonen, Interactional dominance in dyadic communication: A presentation of initiative-response analysis, *Linguistics*, vol.26, no.3, pp.415-442, 1988.

[5] C. Danescu-Niculescu-Mizil and L. Lee, Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs, *Proc. of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, 2011.

[6] Y. D. Prabowo, H. L. H. S. Warnars, W. Budiharto, A. I. Kistijantoro, Y. Heryadi and Lukas, LSTM and simple RNN comparison in the problem of sequence to sequence on conversation data using Bahasa Indonesia, *2018 Indonesian Association for Pattern Recognition International Conference (INAPR)*, Jakarta, Indonesia, pp.51-56, 2018.

[7] C. Olah, *Understanding LSTM Networks*, https://colah.github.io/posts/2015-08-Understanding-LSTMs/, Accessed on April 10, 2020.

[8] M. Heilman and N. A. Smith, Good question! Statistical ranking for question generation, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp.609-617, 2010.

[9] I. V. Serban, A. García-Durán, C. Gulcehre, S. Ahn, S. Chandar, A. Courville and Y. Bengio, Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus, *arXiv Preprint*, arXiv: 1603.06807, 2016.

[10] N. Mostafazadeh, I. Misra, J. Devlin, M. Mitchell, X. He and L. Vanderwende, Generating natural questions about an image, *arXiv Preprint*, arXiv: 1603.06059, 2016.

[11] S. Reddy, D. Chen and C. D. Manning, COQA: A conversational question answering challenge, *Transactions of the Association for Computational Linguistics*, vol.7, pp.249-266, 2019.

[12] K. Xu, Y. Feng, S. Huang and D. Zhao, Hybrid question answering over knowledge base and free text, *Proc. of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp.2397-2407, 2016.

[13] R. Sutoyo, A. Chowanda, A. Kurniati and R. Wongso, Designing an emotionally realistic chatbot framework to enhance its believability with AIML and information states, *Procedia Computer Science*, vol.157, pp.621-628, 2019.

[14] Y. Fu and Y. Feng, Natural answer generation with heterogeneous memory, *Proc. of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp.185-195, 2018.

[15] R. Sutoyo, C. Quix and F. Kastrati, FactRunner: A new system for NLP-based information extraction from Wikipedia, in *Web Information Systems and Technologies. WEBIST 2013. Lecture Notes*

in Business Information Processing, K. H. Krempels and A. Stocker (eds.), Berlin, Heidelberg, Springer, 2014.

[16] R. Sutoyo, C. Quix and F. Kastrati, FactRunner: Fact extraction over Wikipedia, *WEBIST*, pp.423-432, 2013.

[17] A. Bordes, Y.-L. Boureau and J. Weston, Learning end-to-end goal-oriented dialog, *arXiv Preprint*, arXiv: 1605.07683, 2016.

[18] C. Tho, A. S. Setiawan and A. Chowanda, Forming of dyadic conversation dataset for Bahasa Indonesia, *Procedia Computer Science*, vol.135, pp.315-322, 2018.

[19] K. Greff, R. Srivastava, J. Koutník, B. Steunebrink and J. Schmidhuber, LSTM: A search space odyssey, *IEEE Trans. Neural Networks and Learning Systems*, vol.28, no.10, pp.2222-2232, 2016.

[20] I. Sutskever, O. Vinyals and Q. V. Le, Sequence to sequence learning with neural networks, *arXiv Preprint*, arXiv: 1409.3215, 2014.

[21] D. Bahdanau, K. Cho and Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv Preprint*, arXiv: 1409.0473, 2014.

[22] G. Liu and J. Guo, Bidirectional LSTM with attention mechanism and convolutional layer for text classification, *Neurocomputing*, vol.337, pp.325-338, 2019.

[23] S. Santhanam, Context based text-generation using lstm networks, *arXiv Preprint*, arXiv: 2005.00048, 2020.

[24] A. Chowanda and A. D. Chowanda, Generative Indonesian conversation model using recurrent neural network with attention mechanism, *Procedia Computer Science*, vol.135, pp.433-440, 2018.

[25] Md S. Islam, S. S. S. Mousumi, S. Abujar and S. A. Hossain, Sequence-to-sequence Bangla sentence generation with LSTM recurrent neural networks, *Procedia Computer Science*, vol.152, pp.51-58, 2019.

[26] D. Britz, *Understanding Convolutional Neural Networks for NLP*, http://www.wildml.com/2015/11/understanding-convolutional-neuralnetworks-for-nlp/, Accessed on 11/07/2015.

[27] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Prentice Hall, 2008.

[28] W. Zhu, A. Chowanda and M. Valstar, Topic switch models for dialogue management in virtual humans, *The 16th International Conference on Intelligent Virtual Agents (IVA2016)*, pp.407-411, 2016.

[29] J. Weizenbaum et al., ELIZA – A computer program for the study of natural language communication between man and machine, *Communications of the ACM*, vol.9, no.1, pp.36-45, 1966.

[30] N. Bush, R. Wallace, T. Ringate, A. Taylor and J. Baer, Artificial intelligence markup language (AIML) version 1.0.1, *ALICE AI Foundation Working Draft*, 2001.