Enhancing Rasa NLU model for Vietnamese chatbot

Nguyen Thi Mai Trang, Maxim Shcherbakov

***Abstract*—Nowadays, the use of chatbots in industry and education has increased substantially. Building the chatbot system using traditional methods less effective than the applied of machine learning (ML) methods. Before chatbot based on finite-state, rule-base, knowledgebase, etc, but these methods still exist limitation. Recently, thanks to the advancement in natural language processing (NLP) and neural network (NN), conversational AI systems have made significant progress in many tasks such as intent classification, entity extraction, sentiment analysis, etc. In this paper, we implemented a Vietnamese chatbot that is capable of understanding natural language. It can generate responses, take actions to the user and remember the context of the conversation. We used Rasa platform for building chatbot and proposed an approach using custom pipeline for NLU model. In our work, we applied the pre-trained models FastText and multilingual BERT and two custom components for the pipelines. We evaluated and compared our proposed model with existing ones using the pre-defined NLU pipeline. Experimental comparison of three models showed that the proposed model performed better in intent classification and entity extraction.**

***Keywords*—Vietnamese chatbot, RASA NLU, pipeline, Rasa custom components**

I. Introduction

The virtual assistants or chatbots appear more and more in our social life such as Apple Siri, Google Assistant, Amazon Alexa, Microsoft Cortana and Yandex Wordstat. The virtual assistant can help user perform a variety of requests from making a call, searching for information on the internet to booking a ticket, booking a hotel room. Communicating with the virtual assistant via voice or text interaction. Traditional chatbots are built based on rule-based, so when the input questions outside the script, the chatbots will not understand and wait for the customer care staff to respond to the user requests. A chatbot is intelligent or not depends on the ability to understand user context and work independently. To achieve this goal, chatbots must be built on machine learning and artificial intelligence. Last few years, many studies used several machine learning methods based on neural networks [1]-[4] for building conversational AI systems.

With the advantage of machine learning techniques, the chatbot performance increased. The use of chatbots has shown amazing efficiency in many areas of social life. Chabots help businesses save costs, manpower and increase efficiency customer care. On the user side, there are quite a lot of people prefer to interact with chatbots [5]. With the launch of several chatbot platforms such as Facebook Messenger, WhatsApp, Telegram, Skype, Slack the number of chatbots increased day by day. Recently, the use of chatbots increased not only in commercial but also in medicine and education.

Currently, there are many development platforms with supporting for building chatbots based on machine learning such as RASA, Amazon Lex, Microsoft Bot Framework, Google DialogFlow, IBM Watson Assistant, Wit.ai and so on. In this work, we use RASA platform for building our chatbot with some reasons: RASA is an open-source natural language processing tool, it can run locally and has the advantage of self-hosted open source such as adaptability, data control, etc. [6].

The rest of the paper is structured as follows: section II discusses the existing works, section III describes the method to build a Vietnamese chatbot with significant improvement in intent classification and entity extraction, we show and discuss the obtained results in section VI and we close with some concluding remarks and discussion on our future work.

II. Related works

There are many tools/platforms for building a chatbot based on natural language understanding (NLU). We have implemented our research that uses the most common platform is available in the market [7]. Rasa is not only a commercial chatbot building platform but it also greatly aids research. In previous work, we created our chatbot in Russian. But we haven't used a custom pipeline to improve the NLU model. The NLU model is only trained based on supervised embedding, but not applying modern pre-trained models as GloVe, FastText or BERT.

In paper [8], the authors reviewed and analyzed Rasa platform quite in detail. They built a chatbot integrating with API and database. However, the study just built a simple chatbot without using the advanced capabilities of the platform.

In [9], Jiao described the principle of Rasa NLU and designed the functional framework which implemented with Rasa NLU. The author integrated Rasa NLU and NN methods for entity extraction after intent recognition. The study showed that the Rasa NLU outperforms NN inaccuracy for a single experiment.

Recently, Rasa NLU is often used to build a

Manuscript received Oct 15, 2020
Nguyen Thi Mai Trang is with Volgograd State Technical University, Volgograd, RUSSIA, (corresponding author phone: +79667853229; e-mail: m.trang91@gmail.com).
M. Shcherbakov is with Volgograd State Technical University, Volgograd, RUSSIA (e-mail: maxim.shcherbakov@gmail.com).

conversational AI. It comprises loosely coupled modules combining several NLP and ML libraries in a consistent API [10]. In article [11], the author presented the examples for using custom component in Rasa NLU in Vietnamese and Japanese. He created the custom tokenizers for Vietnamese and Japanese and created the custom sentiment analyzer too. The result showed that the NLU model is more suitable and better for Vietnamese and Japanese. The work [12] presented the building of a Vietnamese chatbot using Rasa platform. The author created a custom tokenizer for Vietnamese and trained NLU model by a supervised method.

To the best of our knowledge, there are no studies for building Vietnamese chatbots applying the pre-trained models like FastText, MITIE, BERT in custom Rasa NLU pipeline.

III. METHOD

In this section, we will present the construction of a Rasa chatbot consisting of two main components: Rasa NLU and Rasa Core. We also propose a method to improve Vietnamese chatbot's natural language understanding by using the custom components in pipelines. In addition, the performance of a chatbot also depends on tuning the parameters in the policies that are provided in Rasa Core.

A. RASA flatform

RASA is an open-source implementation for natural language processing (NLU) and Dual Intent and Entity Transformer (DIET) model. RASA is a combination of two modules: Rasa NLU and Rasa Core [13]. Rasa NLU analyses the user's input, then it classifies the user's intent and extracts the entities. Rasa NLU combines different annotators from the spaCy parse to interpret the input data [14].

- Intent classification: Interpreting meaning based on predefined intents. (Example: "How many people are infected with COVID-19 in USA?" is a "request_cases" intent with 97% confidence)
- Entity extraction: recognizing structured data. (Example: USA is a "location")

Rasa Core takes structured input in the form of intents and entities and chooses which action the chatbot should take using a probabilistic model. Fig. 1 shows the high-level architecture of RASA.

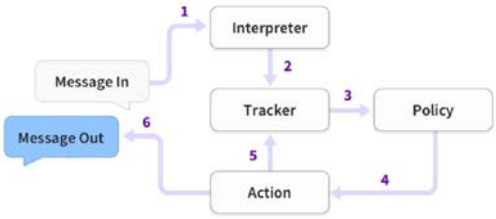


Fig. 1. Diagram of how a Rasa Core app works (from https:/rasa.com/docs/nlu/)

The process of how a Rasa Core app response to a message is explained as follows:

1 - The input message passed to the Interpreter (Rasa NLU). The Interpreter converts user's message into a structured output including the original text, intents and entities.

2 – The Tracker follows conversation state and receives the appearance of the new message.

3- The output of the Tracker passes into Policy, which receives the current state of the tracker

4 - The next action is chosen by the policy

5 – The tracker logs the chosen action

6 – The response is sent to the user, using the pre-defined utterance in nlu.md

B. Processing Rasa pipeline

The process of the input message includes different components. These components are executed sequentially in a processing pipeline. The component processes the input and gives an output which can be used by any following components in the pipeline. A processing pipeline defines which processing stages the input messages will have to pass. The processing stage can be a tokenizer, featurizer, named entity recognizer, intent classifier.

RASA provides pre-configured pipelines which can be used by setting the configuration values as spacy_sklearn, mitie, mitie_sklearn, keyword, tensorflow_embedding. Besides, we can create a custom pipeline by passing the components to the NLU pipeline configuration variable. Fig 2 shows the schema of intent classification.

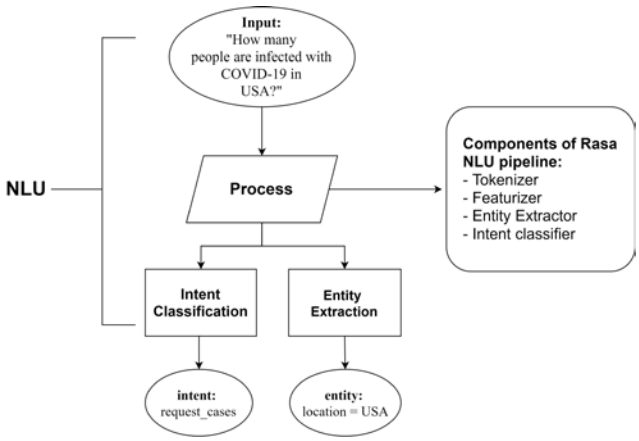


Fig. 2. Schema of intent classification and entity extraction using Rasa NLU.

To build a good NLU model for chatbots, we enhance the model with our own custom components such as sentiment analyzer, tokenizer, spell checker etc.

C. Custom component

Components make up the NLU pipeline. They work sequentially to handle the user's input text into a structured output. In this work, we create two custom components: Vietnamese tokenizer and FastText featurizer which Rasa NLU doesn't currently offer.

A Vietnamese tokenizer is created by using Vietnamese Tookit Underthesea [15]. The FastText featurizer is a dense featurizer which helps to load the FastText embeddings. The implementation of the featurizer requires a tokenizer that is presented in the pipeline (see Fig 4.).

D. Choosing Rasa NLU pipelines

Rasa has pre-configured pipelines which we review for building the chatbot: TensorFlow-based pipeline, ConveRT

pipeline, BERT-based pipeline.

TensorFlow-based pipeline can be used to build the chatbot from scratch. It doesn't use pre-trained word vectors and supports any language that can be tokenized. When we use a custom tokenizer for specify-language, we can replace the "tokenizer_whitespace" with our tokenizer with more accurate. Fig. 3 shows an alternative of Vietnamese tokenizer in TensorFlow-based pipeline.

```
language: "vi"
pipeline:
- name: "VietnameseTokenizer" #"tokenizer_whitespace"
- name: "ner_crf"
- name: "intent_featurizer_count_vectors"
- name: "intent_classifier_tensorflow_embedding"
```

Fig. 3. Example of using Vietnamese tokenizer in Tensorflow pipeline.

The custom pipeline is a list of the names of the components which we want to use. We propose a custom pipeline (cpFastText) using Vietnamese tokenizer and FastText featurizer for loading pre-trained word-embedding in Vietnamese from FastText. Rasa doesn't natively support FastText, so we need to create a custom featurizer for FastText and add into pipeline. We can download the pre-trained model for Vietnamese from FastText [16]. The model with 4.19 Gb in binary compression. Fig. 4 shows the custom pipeline using FastText model for Vietnamese.

```
language: vi
pipeline:
- name: VietnameseTokenizer
- name: LexicalSyntacticFeaturizer
- name: CountVectorsFeaturizer
  analyzer: char_wb
  min_ngram: 1
  max_ngram: 4
- name: fasttext_featurizer.FastTextFeaturizer
  cache_dir: "./Fasttext/vector/cc.vi.300.bin"
  file: cc.vi.300.bin
```

Fig. 4. The custom pipeline using Vietnamese Tokenizer, FastText featurizer and FastText model.

ConveRT pipeline is a template pipeline that uses a ConveRT model to extract pre-trained sentence embeddings. ConveRT pipeline shown the effectiveness of big training data [17]. In this work, we do not use this pipeline to build the Vietnamese chatbot because it is only able to support English.

We also consider a pipeline using the state-of-the-art language model BERT. Rasa provides the pipeline with the configuration for BERT using Hugging Face model. It can be configured with a BERT model inside the pipeline. The example of the BERT pipeline is shown in Fig. 5.

```
pipeline:
  - name: HFTransformersNLP
    model: "bert-base-multilingual-cased"
  - name: LanguageModelTokenizer
  - name: LanguageModelFeaturizer
  - name: CountVectorsFeaturizer
  - name: DIETClassifier
```

Fig. 5. An example of a pipeline using multilingual BERT model

We will implement the experiment and compare the above pipelines in section IV

E. Policies

Rasa Core provides a class rasa.core.policies.policy. It decides the action of chatbot. There are ruse-based and machine-learning policies are detailed in table 1.

TABLE I
LIST OF POLICIES

Type Policy	Name policy	Featurization
Machine learning policy	TED policy – the transformer embedding dialogue	The policy concatenates the user input, system actions and slots
	Memoization Policy	The policy remembers the stories from the training data. It checks the matching story of the current conversation and predicts the next action from the matching story with the confidence of [0,1]. Number of turns of the conversation is indicated in *max_history*
	Augmented Memoization Policy	It remembers examples from matching story for up to *max_history* turns. It is similar to the Memoization Policy. In addition, the policy has a forgetting mechanism.
Rule-based Policies	Rule policy	A policy handles conversation parts that follow a fixed behavior and makes predictions using rules that have been in the training data.
Configuring Policies	Max History	It controls the number of dialogue history that model looks at to predict the next action.
	Data Augmentation	It determines how many augmented stories are subsampled during training.
	Featurizers	It provides to apply machine learning algorithms to build up vector representations of conversational AI.

The policies are configured in config.yml. Two

parameters Max_History and Data Augmentation affect the performance of the model [8]. The policy is used affects the performance of the model. So we need to review and tune the parameters of the policies and be able to use the polices in tandem.

IV. Experiment

A. Experimental setup

In this work, we experimented in a dataset that includes 40 intents, 8 entities and 1000 examples. The intents and utterances are stored in nlu.md. The stories of conversations are presented in stories.md. The pipeline and policies are in config.yml. The domain.yml file defines the domain in which chatbot operates. The domain specifies the intents, entities, slots, utterance responses, actions and a configuration for conversation sessions. We specify the expiration time of a conversation session is 60 seconds.

 We created two custom components (Vietnamese tokenizer, FastText featurizer) as described in section III .

We used three pipelines for evaluation were TensorFlow embedding, cpFastText pipeline and BERT pipeline. In cpFastText, a Vietnamese tokenizer and a FastText featurizer are added into the pipeline. In BERT pipeline (mBERT), we configured it with the BERT multilingual base model (cased) that pre-trained on the top 104 languages with the Wikipedia dataset. Therefore, we will compare the proposed cpFastText model with the two basic models (Tensorflow and mBERT) that are configured using existing components from Rasa NLU.

B. Experimental results

Evaluating Rasa NLU models based on three metrics: precision (1), F1-score (3) and accuracy (4). Correctly predicted observations (True Positives) is the number of observations that were predicted correctly for the class. They belonged to the class and the model classified them correct. We denote the True Positive as TP, the True Negative as TN, the False Positive as FP and the False Negative as FN. Then,

$$\Pr ecision = \frac{TP}{TP + FP} \quad (1)$$

$$\mathrm{Re}\, call = \frac{TP}{TP + FN} \quad (2)$$

$$F1 = 2.\frac{\Pr ecision.\mathrm{Re}\, call}{\Pr ecision + \mathrm{Re}\, call} \quad (3)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

We evaluated the intent classification by F1-score, accuracy and precision using cross-validation. The results on three models in table 2 show the micro-average scores on the test set for intent classification and entity extraction over 5-fold cross-validation. As the results, the proposed cpFastText model is the best model for intent recognition (F1-score = 0.863, accuracy = 0.865 , precision = 0.867) and entity extraction (F1-score = 0.815, accuracy = 0.997, precision = 0.864). The mBERT model gives the worst results. It can be explained that, because the data set is not large enough (1000 examples) and applying the mBERT model is not appropriate, instead of using the TensorFlow-based model gives better results.

TABLE II
The Comparison Of Three NLU Models For Intent Classification And Entity Extraction

Model	Metric	Intent	Entities
TensorFlow	F1-score	0.826 ± 0.033	0.668 ± 0.185
	Accuracy	0.827 ± 0.032	0.996 ± 0.002
	Precision	0.849 ± 0.029	0.681 ± 0.174
mBERT	F1-score	0.607 ± 0.039	0.729 ± 0.127
	Accuracy	0.641 ± 0.033	0.996 ± 0.002
	Precision	0.647 ± 0.044	0.762 ± 0.110
cpFastText	F1-score	$\mathbf{0.863 \pm 0.010}$	$\mathbf{0.815 \pm 0.042}$
	Accuracy	0.865 ± 0.011	0.997 ± 0.001
	Precision	0.876 ± 0.011	0.864 ± 0.072

To compare the multiple pipelines we used Rasa with the command as follow:

rasa test nlu --config Tensorflow.yml cpFastText.yml mBERT.yml
--nlu data/nlu.md --runs 3 --percentages 0 25 50 75 85

Then the models are evaluated on the test set and the F1-score for each exclusion percentage is recorded. The comparison of three Rasa pipelines presents in Fig. 6. The Fig. 6 shows that cpFastText is the pipeline with the best F1-score.

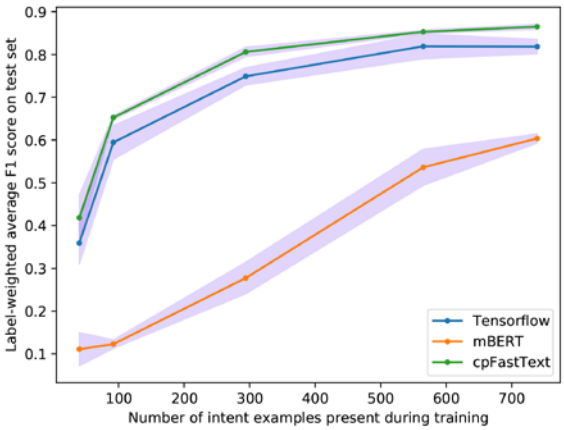
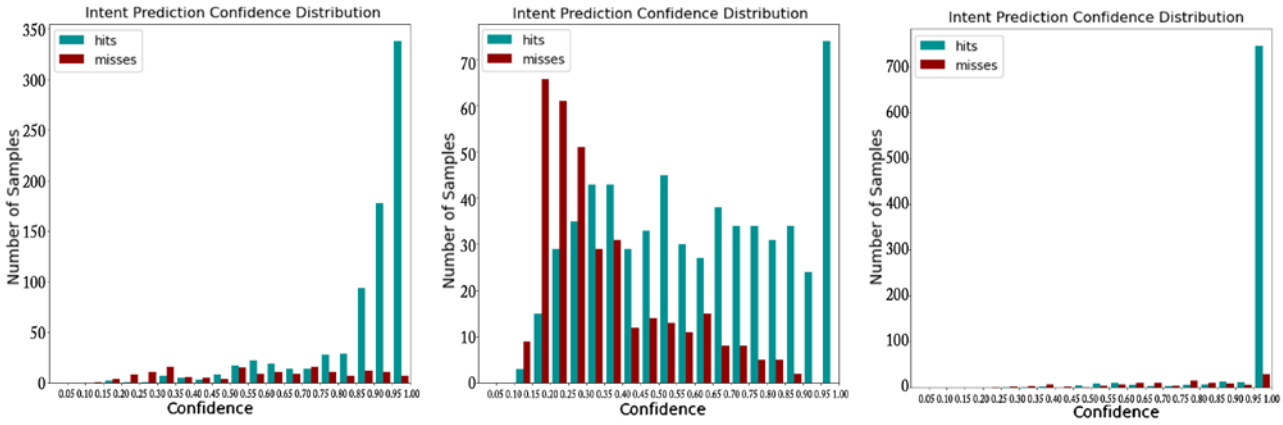


Fig. 6. Comparison of Rasa NLU pipelines.

We created plots of intent prediction confidence distribution for three models (see Fig. 7). The plot with two columns per bin showing the confidence distribution for hits and misses. The plot of cpFastText model shows the most correct prediction confidences. Thus, the results show that the models cpFastText using the proposed method have given the best result for intent classification. So we have chosen the proposed cpFastText model to build the chatbot.

Then we evaluated the dialogue model using Rasa Core. For cpFastText model, we have used TED Policy, Memoization Policy, Fallback Policy and set max history to 5, nlu threshold to 0.7, core threshold to 0.5.

The confusion matrix of the actions presents in Fig. 8. Evaluation on conversation level and action level on three metrics F1-score, accuracy and precision. The results are presented in table 3.



a) Tensorflow model b) mBERT model c) cpFastText model

Fig. 7. Intent prediction confidence distribution of three models.

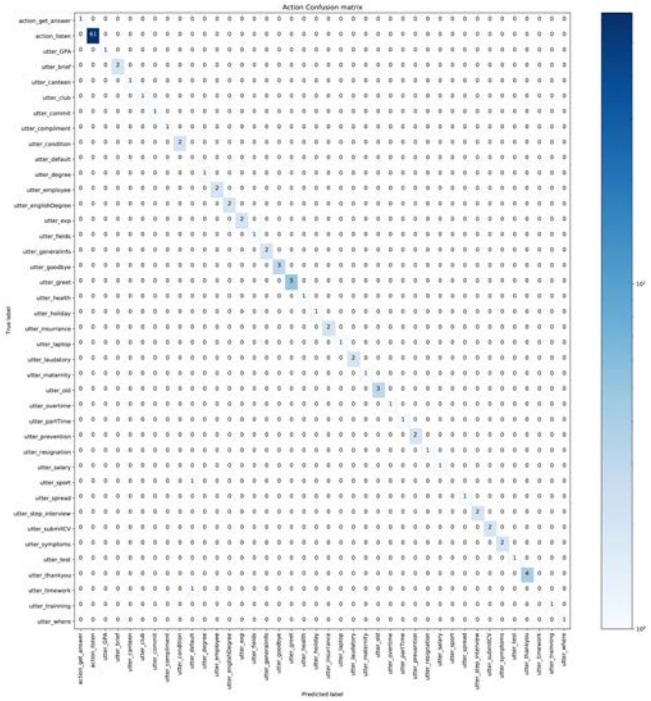


Fig. 8. Confusion matrix of the actions.

TABLE III
EVALUATION OF DIALOGUE MODEL

Metric	Evaluation on conversation level	Evaluation on action level
F1-score	0.976	0.984
Accuracy	0.952	0.984
Precision	1.000	0.984

V. CONCLUSION

In this paper, we presented the method to improve the performance of a chatbot using custom Rasa NLU pipeline. Using custom components is appropriate for NLU model in non-English languages. As the results, the proposed model showed the best result in intent classification and entity extraction. The proposed cpFastText model performed better compared to the TensorFlow-based (F1-score is 3.7% higher), mBERT (F1-score is 25.6%). The dialogue model Rasa Core got an accuracy of 95.2% on conversation level and 98.4% on action level.

The future scope of this study, we build a model for Vietnamese spell checker based on neural machine translation and use it as a new custom component in NLU pipeline. Besides, we will combine our chatbot with a question answering system based on BERT model for answering the non-predefined questions.

REFERENCES

[1] T. Nguyen and M. Shcherbakov, “A Neural Network based Vietnamese Chatbot,” in *2018 International Conference on System Modeling & Advancement in Research Trends (SMART)*, 2018.

[2] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, “Building end-to-end dialogue systems using generative hierarchical neural network models,” *arXiv [cs.CL]*, 2015.

[3] D. Al-Ghadhban and N. Al-Twairesh, “Nabiha: An Arabic dialect chatbot,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 3, 2020.

[4] T.-H. Wen *et al.*, “A network-based end-to-end trainable task-oriented dialogue system,” in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017.

[5] “Need-to-know chatbot statistics in 2020,” *Chatbot.com*. [Online]. Available: https://www.chatbot.com/blog/chatbot-statistics/. [Accessed: 06-Oct-2020].

[6] D. Braun, A. Hernandez-Mendez, F. Matthes, and M. Langen, “Evaluating natural language understanding services for conversational question answering systems,” in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, 2017.

[7] T.M.T. Nguyen, M.V. Shcherbakov, “Целевой чат-бот на основе машиного обучения [A goal-oriented chatbot based on machine learning].” Modeling, optimization and information technology, May 2020. [Online] Available: https://moit.vivt.ru/wp-content/uploads/2020/05/NguyenShcherbakov_2_20_1.pdf

[8] R. K. Sharma and National Informatic Center, “An Analytical Study and Review of open source Chatbot framework, Rasa,” *Int. J. Eng. Res. Technol. (Ahmedabad)*, vol. V9, no. 06, 2020.
[9] A. Jiao, “An intelligent chatbot system based on entity extraction using RASA NLU and neural network,” *J. Phys. Conf. Ser.*, vol. 1487, p. 012014, 2020.
[10] T. Bocklisch, J. Faulkner, N. Pawlowski, and A. Nichol, “Rasa: Open Source Language Understanding and Dialogue Management,” *arXiv [cs.CL]*, 2017.
[11] P. H. Quang, “Rasa chatbot: Tăng khả năng chatbot với custom component và custom tokenization(tiếng Việt tiếng Nhật),” Viblo, 16-Mar-2020. [Online]. Available: https://viblo.asia/p/rasa-chatbot-tang-kha-nang-chatbot-voi-custom-component-va-custom-tokenizationtieng-viet-tieng-nhat-Qbq5QN4mKD8. [Accessed: 14-Sep-2020]
[12] M. V. Do, “Xây dựng chatbot bán hàng dựa trên mô hình sinh,” M.S. thesis, Graduate Univ. of Sc. and Tech., Hanoi, 2020. Accessed on: 10 Sep, 2020. [Online]. Available: http://gust.edu.vn/media/27/uftai-ve-tai-day27665.pdf
[13] “The Rasa Core dialogue engine,” *Rasa.com*. [Online]. Available: https://legacy-docs.rasa.com/docs/core/. [Accessed: 1-Oct-2020].
[14] H. Agarwala, R. Becker, M. Fatima, L. Riediger, “Development of an artificial conversation entity for continuous learning and adaption to user’s preferences and behavior” [Online]. Available: https://www.di-lab.tum.de/fileadmin/w00byz/www/Horvath_Final_Documentation_WS18.pdf. [Accessed: 25-Sep-2020].
[15] “underthesea,” *PyPI*. [Online]. Available: https://pypi.org/project/underthesea/. [Accessed: 25-Sep-2020].
[16] “Word vectors for 157 languages fastText,” *Fasttext.cc*. [Online]. Available: https://fasttext.cc/docs/en/crawl-vectors.html. [Accessed: 23-Sep-2020].
[17] A. Singh, “Evaluating the new ConveRT pipeline introduced by RASA,” *Medium*, 03-Dec-2019. [Online]. Available: https://medium.com/@arunsingh_19834/evaluating-the-new-convert-pipeline-introduced-by-rasa-3db377b8961d. [Accessed: 30-Aug-2020].

Nguyen Thi Mai Trang,
PhD student in CAD Department, Volgograd State Technical University, Volgograd, Russia
Email: m.trang91@gmail.com

Maxim Shcherbakov,
Dr. Tech. Sc., Head of CAD Department, Volgograd State Technical University, Volgograd, Russia
Email: maxim.shcherbakov@gmail.com