POLLACK PERIODICA
An International Journal for Engineering and Information Sciences
DOI: 10.1556/606.2018.13.3.11
Vol. 13, No. 3, pp. 107–118 (2018)
www.akademiai.com

SHORT TEXT EVALUATION WITH NEURAL NETWORK

[1] Ádám PINTÉR, [2] Balázs SCHMUCK, [3] Sándor SZÉNÁSI

[1] Doctoral School of Applied Informatics and Applied Mathematics, Óbuda University
Bécsi út 96/b, H-1034 Budapest, Hungary, e-mail: pinter.adam@nik.uni-obuda.hu
[2,3] John von Neumann Faculty of Informatics, Óbuda University
Bécsi út 96/b, H-1034 Budapest, Hungary
e-mail: [2]schmuck.balazs@nik.uni-obuda.hu, [3]szenasi.sandor@nik.uni-obuda.hu

Received 31 December 2017; accepted 30 May 2018

Abstract: The aim of this paper is to present a technique, which uses machine learning to process the short text answers with Hungarian language. The processing is based on a special neural network, the convolutional neural network, which can efficiently process short text answer. To achieve precise classification for training and recall grammatically consistent answers and the conversion of the text to the input are inevitable. To convert the input, continuous bag of words and Skip-Gram models will be used, resulting in a model that will be able to evaluate the Hungarian short text answers.

Keywords: Short text evaluation, Neural network, Convolutional neural network

1. Introduction

In this paper, a method which is based on convolutional neural network for processing Hungarian short text answers is presented, including the mapping of the input and the spelling check and correction. Test results of the proposed method and a comparison with the results of the static method used by the Short Text Response Module (STRM) of the eMax (intelligent assessment system for e-learning) system of Óbuda University [1].

1.1. Overview of task and their classification types

Task types can be divided into two large groups based on the answer to the question:

- Passive task types;
- Active task types.

For passive type tasks the answers are predefined, the student, answering the question, can choose one or more from the answer list. Algorithmic processing is recognition; algorithm compares student response with a set of good answers. For active type tasks, the answer can be defined in free formulation by the student. In this case, only the syntax of the answer can be inferred, its type and form cannot be prepared (for example, mathematical formulas). In this case, the student's answer is represented as an open set, which can be evaluated by a recalling algorithm.

Active type tasks include a group of natural language task that include four subtasks:

- Fill-the-gap;
- Listing;
- Short text answer;
- Essay.

Fill-the-gap tasks are typically used in grammatical tests, where the answer is restricted to word(s), part of word(s), and sets of abbreviations. This type of tasks can be evaluated by pattern matching, considering the possibility of spelling errors.

Listing tasks is where the student lists the answers in a free order and formulation instead of typing the words in the predetermined locations. In that case the answer is not necessarily just a single word. Evaluation of listing task can be traced back to pattern matching since there is no need to examine the interrelation between the listed items.

In case of short text answer and essay type tasks, the student may give a free text answer with his own words to the question asked. One of the methods of their automated processing is keyword search technique where the answer is marked good if the keywords exist in the answer, otherwise false. This method may cause false positive results because the relationships between the words can modify the meaning of the sentence. Therefore, this solution cannot be used in several cases. However, keyword based techniques can be a good starting point for processing, complemented by the functionality of understanding and mapping the sentence, the evaluation can achieve good results.

To identify a short text answer, it is essential to define the differences between the natural language subtasks. The fill-the-gap and the listing type tasks are easily recognizable by the nature of the question, but the boundaries of the essay and the short text answers are fuzzy. To determine the differences between subtask types, the separation of the following properties may help:

- Length;
- Focus;
- Openness.

Table I summarizes the properties and subtasks types' relationships.

The length property is limited to a single word or phrase for fill-the-gap and listing answers, while in the essay may exceed the plurality of pages from a few paragraphs.

Between that two, there is a short text answer, where the length of the answer should be from one word to a paragraph.

Table I

Properties of natural language tasks

	Fill-the-gap/Listing	Short Text Answer	Essay
Length	word(s), phrase(s)	from a few words up to a paragraph	one or two paragraphs up to a few pages
Focus	word(s) and synonyms	content	style
Openness	fixed	closed	open

The focus determines which of the elements and attributes of the answer are evaluated during the processing. In this case, in the short text answers the actual content comes to the fore, while in the essay the focus is more on the writing style than on the concise formulation of professional content. For short text answer, focus can be narrowed or expanded by expecting or excluding certain words.

The last feature characterizes the question's openness. A question is considered open, if the answer contains examples and opinions, in which case the unknown notion can be paraphrase. The question is closed, when the answer relies solely on the facts, contains statements that are short, meaningful and not lengthy. It follows from the two definitions that the short text answers are closed, that is, they are objective, contain no unnecessary descriptions, and the essays are open, formulating examples and opinions.

These properties facilitate the evaluation of short text answers, but it must be borne in mind that students will not necessarily follow these properties in their own answers [2].

2. Short text answer evaluation with convolutional neural network

Automatic processing methods for short text responses can be divided into two categories: *Keyword-based method and content recognition*. The method chosen will greatly affect the structure and the operation of the system that has been developed.

With the keyword-based method, the creator of the question must specify the set of keywords that contains the correct, wrong, and expected keywords. The system will be evaluated the student's answer by these keywords with pattern matching, considering the abbreviations and the possibility of spelling error. The most complex of these techniques are when not only the number of mistyped characters are maximized in word (depending of the length of the word), but also the difference in the length of the word affects the process of correction. In general answers cleaning and correcting is done using a dictionary of word frequency, complemented by algorithms that are able to detect the spelling errors. To use these dictionaries, it is required to generate all permutations of the input word characters, from which the dictionary will determine by frequency what the 'most appropriate' word is. Unfortunately, the use of this method is extremely costly, and the result is not guaranteed, especially in cases where the input

word is not in the dictionary. The acceptable solution in this case is to expand the word frequency dictionary with additional words.

During content recognition, the underlying meaning and content of the answer are attempted to be evaluated, considering the possibility of spelling errors, although these mistakes less affect the processing. In this case, the answer is not a set of words that need to be processed by elements, but they are nodes of a closely related and interconnecting graph. So, the algorithm needs to try to understand the answer, since the sequence order of words (or clauses, sentences) has effect and modify the sentence concept.

The two methods can be used alone to the evaluation of short text answers, but it is also possible to combine the two methods. Regardless of which model is selected, the following four important aspects are required to evaluate the short text answer:

- definition of keywords and synonyms;
- evaluating the relationships between keywords;
- recognition of text modifying effects;
- compare the evaluated student's answer with the instructor’s answers.

2.1. Model of the system

The method is a type of content recognition based on the convolution neural network, which is complemented by data set preparation, pre-processing, and model efficiency examination, as it is shown in *Fig. 1*. In this figure, blue dashed labeled elements only used in the training phase and the elements marked with green dotted are only used to determine the model’s efficiency.

During content recognition, the underlying meaning and content of the answer are attempted to be evaluated, considering the possibility of spelling errors, although these mistakes less affect the processing. In this case, the answer is not a set of words that need to be processed by elements, but they are nodes of a closely related and interconnecting graph. So, the algorithm needs to try to understand the answer, since the sequence order of words (or clauses, sentences) has effect and modify the sentence concept.

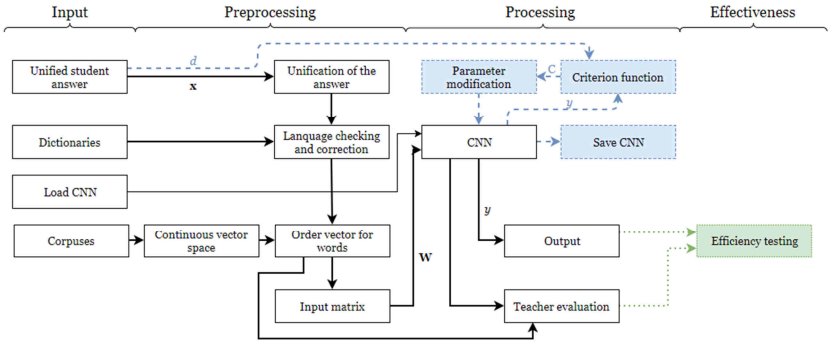

Fig. 1. The processing model of short text answers

2.2. Input

The most important element of the neural network [3], [4] is a set of consistent training answers that can be used to set up the network's expected operation. In addition to the set of training answers, there will be a need for a set of validation answer to check the correctness of the training process, avoiding over-learning and over-fitting. Both sets contain mixed positive and negative patterns, including the expected output, but when using the model, the input answers will not contain the output values, they will be produced by the model and, if necessary, the instructor can check it.

The input conversion is part of the process that converts student answers into a unified form of data storage. The written answers' (which can came from a paper-based questionnaire or an electronic test system) uniformization is required that both machine and human easily understand. The selected storage method is a format defined by Comma Separated Values (CSV), where the data is organized in separate rows, within the individual values are separated by commas. The values of the data in a row are the student's text answer and the corresponding score (given by the instructor or the trained model). The stored data set will have a special case, the question, which will always be the first element of the data set. The advantage of storage is that its algorithmic processing can be easily done, thanks to the separating characters, and the file can be read and edited independently of the model.

2.3. Data pre-processing

The essential issue of processing natural language with neural networks is the form of input data. For example, take the next sentence, which is wanted to process:

'Use the FIFO policy if the item is perishable, otherwise use the LIFO policy.'

In the example, in addition to the correct words, there are punctuation, abbreviations and incorrect words. If this answer is evaluated without pre-processing, the network will not be able to evaluate correctly, because the model cannot pair the words to vectors due to incorrect words. Additional problems to be prepared for:

- If the word is preceded or followed by punctuation (uniformization required);
- If the word is mistyped (language correction required);
- If the vector space does not include the word (reconstruction of a vector space required);
- If the word has another form in the answer (additional dictionaries and corpus are required).

In addition to the listed problems, there may be additional unexpected events during the model's knowledge recall phase (for example the case of uppercase character), which the model must be prepared for. The transformation of the students' answer must be performed carefully. If the system is inflexible, then in these cases the vectors for the words will not be found (since they will not exist), so the processing of the answer is interrupted. However, if the model is going to be too flexible (for example, the algorithm accepts the word '*first-in FO*' instead of '*FIFO*'), the student's answer during the pre-processing will be damaged (or the wrong answer can be accepted).

When unifying the response, the unnecessary punctuations are removed, which will reduce the size of the resulting vector space. Examining the meaning of the example it can be seen, it is irrelevant to the meaning of the sentence that the comma appears after the word in the sentence: '*perishable*', or not: '*perishable*'. In contrast to the example above, in the vector space model, both types of words should be trained. The true meaning of the words will be determined by the context of the words that Convolutional Neural Network (CNN) will learn to recognize and interpret, where these differences will no longer matter [5].

A critical point to solve this problem (word vector mapping) is to check the grammaticality of the words. In addition, to the incorrectly written words resulting in meaningless sentences, they will not be found in the vector space, so algorithmically the processing will fail. Spell checking, and correction is based on the repair mechanism used by Google translate. Feature of the method is the frequency dictionary, which contains words and their average appearance. The following frequency dictionaries are used in the model:

- general frequency dictionary that contains the frequency of the general words of the language;
- additional frequency dictionary, which contains words specific to the task/speciality;
- synonym dictionary, which contains the synonyms of each word (in many cases, for example, it is not interesting to use the term in 'first in first out' or in 'FIFO' format in the answer, but it is preferable to avoid redundancies from the processing point of view);
- a special symbol list that contains symbols that must be deleted from the response before processing the words (for example: *).

During the correction, the (formatted, punctuated) answer is split down into words, and then, as a first step, the terms are unified according to the synonym dictionary. After that, the processing will go through all the words in the answer and verify whether it is in the word frequency dictionary (both general and additional) or not. If it is found, it is assumed that the word is in correct format and its processing is over. If it cannot be found, new words will be created from characters in the word by transposing, permuting, supplementing, or deleting characters. At the end of this generation, the generated word set is sorted by using the frequency dictionary and the word is selected with the highest value from the set. If that process fails, then the original word remains untouched in the answer [6].

The disadvantage of this method is for example, if the frequency dictionary does not contain a word, but contains a similar one, then the original correct word will be replaced by a wrong one during the reparation.

Spell checking and correction creates a sentence that does not distort the original meaning of sentence and is suitable for neural network processing.

The condition of making a good vector space is, to use the relevant corpuses, which, in addition to having all the relevant words, also represents the appropriate quantity and quality relationships. In the case of corpuses, the process of uniformization is also recommended, thus reducing the size of the vector space [5].

The vector space was constructed with word2vec, where the Continuous Bag Of Words (CBOW) and the Skip-Gram vector space building techniques are available. [7] To build the space CBOW is used because the existing corpuses were a smaller set [8] (for larger corpus the space created by the Skip-Gram model was better). Finally, the vector space is created (with default size of 100), which can be used to convert the student's text answers to the matrix. This matrix can be loaded to the neural network input.

To create the correct matrix, the student answer's every word must be found in the vector space, which will be a simple link between words and vectors. A problem arises when word or words are not found in the space. This may happen if the uniformization of the student's answer was incorrect or if a word appeared in the answer that was not contained by the corpus. In this case, the options will be the following:

- That word is ignored, knowing that the answer will be distorted, or it may be meaningless (not recommended);
- The answer will be forwarded to instructor evaluation without any further processing (recommended for the trained system during the recall phase);
- If the word was meaningful, the corpus will be reviewed, the vector space, and the process of uniformization the response (recommended under training of model).

Once the corresponding vectors are found for the words of the answer, the matrix can be produced, the lines being the representations of the words' vectors, and the columns being the elements of the vectors. The length of the vectors, i.e. the columns of the matrix, can be determined in the construction of the vector space. However, the number of words, i.e. the lines of the matrix, must be fixed in advance, resulting in either having to fill the remaining lines with vectors that do not have meaning, or the longer answers must be cut off. Based on the experience so far (and the nature of the short textual response), it can be stated that 100 words (the number of matrix rows) should be sufficient to formulate the answer, which value can be set depending on the task and the result of the training.

In the case of generated matrix (with fixed size of 100 rows and 100 columns), the storage may be questionable, which is not necessary because the information contained therein can be found in the student response and the vector space. Once we have been able to generate the input matrix, we can start evaluated it by CNN.

2.4. Grading

Processing can be divided into two processes, depending on whether the aim is to train or use the network. If the aim is the training, the input data needs to be separated into training and validation set, and the parameters for the network must be set based on the criterion function and instructor evaluation. If an already trained neural network is wanted to be used, it must be loaded it into the model, and then pass the answers to get the output values (the evaluation process will update the input CSV file content with the score of the answers).

When selecting elements of a training data set, it may be advisable to proceed as follows:

- answers differ by nature as far as possible;
- most (but not all) words appear in the selected set;
- selected set should contain almost the same number of negative and positive samples;
- at least 50% and up to 75% of all answers should be selected.

In processing of natural language, CNN uses convolutional filters that typically cover the entire line (the whole word) in the matrix, unlike image processing, where the filters cover two or three pixels. That is, the width of the filter generally corresponds to the width of the matrix and its height varies, typically two to five words as it can be seen in *Fig. 2*, [9], [10].

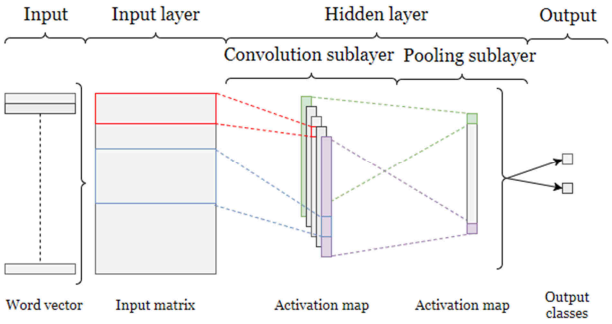

Fig. 2. Structure of CNN in natural language processing

Intuitions useful for machine vision (invariance, composition) will not be useful when processing natural language. The composition of the words can be decisive at low levels (verbs, terminations, etc.), but the higher abstraction level will not help to processing, unlike image processing. Therefore, during processing, we concentrate on the presence or the place of occurrence of words.

The CNN network contains two hidden layers. The first one is the convolution layer, which task is to highlight the essential information in the answer. The second one is the pooling layer, which aggregate the convolution layer output into a single value.

In the convolution layer, the following formula is used to determine the *i*-th output:

$$\mathbf{C}_i = \sum_{j=1}^{m} \sum_{k=1}^{n=100} \left(\mathbf{S}[i-m+1:i] \circ \mathbf{F}\right)_{j,k}, \tag{1}$$

where **S** is the input matrix (part of the answer, 2 to 5 number of words) with zero padded top and bottom, [i-m+1:i] is the matrix restriction between the row *i-m+1* and *i*; **F** is the filter matrix with height *m*; ∘ operation is the Hadamard entrywise product of the matrices; $\mathbf{C}_i$ is the *i*-th component of the output with the given filter size. This operation works similar as convolution operation.

The pooling layer was chosen as the max pooling because it does not bring another parameter into the model. However, it has a disadvantage that network can easily overlearn itself. The following general formula was used in the layer:

$$\mathbf{C}_{pooled} = \begin{bmatrix} pool\left(\alpha\left(\mathbf{C}_1 + b_1 \cdot \mathbf{e}\right)\right) \\ \cdots \\ pool\left(\alpha\left(\mathbf{C}_n + b_n \cdot \mathbf{e}\right)\right) \end{bmatrix}, \tag{2}$$

where pool is the max pooling operation; $\alpha()$ is the activation function (threshold of the pooling); $\mathbf{C}_i$ is the i-th component of the output of the convolution, for which the b_i bias value multiplied by the **e** unit vector is added, $\mathbf{C}_{pooled}$ is the pooling output (in this case a simple number).

Finally, the network contains fully connected layer, which one is a simple back propagated neural network with SoftMax activation function. This layer will classify the pooled information by probability distribution:

$$p\left(y = j \mid \mathbf{x}\right) = \frac{e^{\mathbf{x}^T \boldsymbol{\theta}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^T \boldsymbol{\theta}_k}}, \tag{3}$$

where $\boldsymbol{\theta}$ is the weight vector of the k-th class, **x** is a final abstract representation of the input example. Output of the formula will be a vector with as many elements as the number of classes to be created and their values will be the probability of the class assignment [11], [12].

For each answer will have a 2-element output vector (the good and the bad answer probability). It must be determined whether the response was good or bad based on the output vector. The vast majority of input responses will not be clearly classified into one class, so one must compromise when accepting the answer as good or bad. For example, let's suppose that there is a trained network that classifies 37% of the input answers in the 'bad response' class and 51% in the 'good response' class. The model at present is more of a penalty than a permissive, so the answer is good if it is at least 75% part of the good class and is not part of the bad class by more than 25%.

2.5. Effectiveness

When examining the effectiveness of the model, it is assumed that the network will give the correct answer to each question. That is, on the model, if all available data is passed, after the required dictionaries, the vector space, and if the trained model are loaded. At the end of the evaluation, the model will mark the answers by the following categories:

- Instructor evaluation required;
- The answer to the question is correct;
- The answer to the question is incorrect.

The resulting values are compared with the results provided by the instructor, which determines how many responses were correctly or incorrectly evaluated, and how many

could not be evaluated. This will result the percentage of the model that will characterize processing efficiency.

3. Overall results

To test the model, English SMS are selected that were classified into to two categories based on their content (ham, spam), and English Forum posts, which were also separated into two group based on their emotional content (positive, negative). In both cases, nearly 50% of the answers were selected to train the model. The English dataset evaluation results are summarized in *Table II* and in *Fig. 3*.

Table II

Results of the English data sets evaluated by the model

	Forum posts		SMS		
Number of items in data set	1000	100%	1324	100%	100%
Answer referenced to 'Instructor Verification' (due to matrix mapping)	519	51.9%	604	45.6%	48%
Answer suitable for training	481	48.1% (100%)	720	54.4% (100%)	52% (100%)
Correctly evaluated data	84%	621	86%	86%	85%
Incorrectly evaluated data	16%	99	14%	14%	15%
Training date set size	231	100%	364	100%	100%
Correctly evaluated data	97%	318	87%	87%	91%
Incorrectly evaluated data	71%	46	13%	13%	9%
Validation data set size	250	100%	356	100%	100%
Correctly evaluated data	71%	303	85%	85%	79%
Incorrectly evaluated data	29%	53	15%	15%	21%

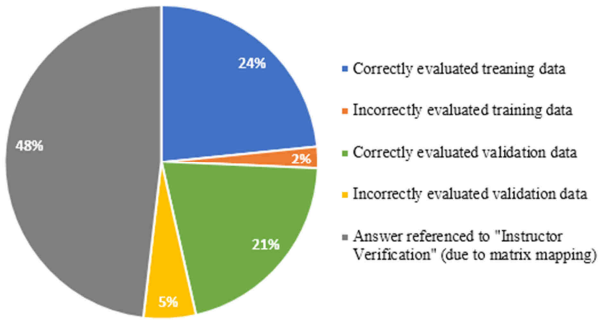

Fig. 3. The summarized results of the English data set evaluated by the model

After that the model is tested with Hungarian students' answers, which were evaluated by the instructor. This data set size is much smaller, so the evaluation was executed again on a reduced set of the English SMS data set to be able to compare the differences. The evaluation results are summarized in *Table III* and in *Fig. 4*.

Table III

Results of the reduced SMS English data set and the Hungarian students' answers evaluated by the model

	SMS (reduced size)		Student's answers		
Number of items in data set	150	100%	82	100%	100%
Answer referenced to 'Instructor Verification' (due to matrix mapping)	42	28%	11	13.5%	20.8%
Answer suitable for training	108	72% (100%)	71	86.5% (100%)	79.2% (100%)
Correctly evaluated data	73	67.5%	50	70.4%	69%
Incorrectly evaluated date	35	32.5%	21	29.6%	31%
Training date set size	60	100%	39	100%	100%
Correctly evaluated training answers	49	81.6%	31	79.5%	81%
Incorrectly evaluated training answers	11	18.4%	8	20.5%	19%
Validation data set size	48	100%	32	100%	100%
Correctly evaluated validation answers	24	50%	19	59.3%	55%
Incorrectly evaluated validation answers	24	50%	13	40.6%	45%

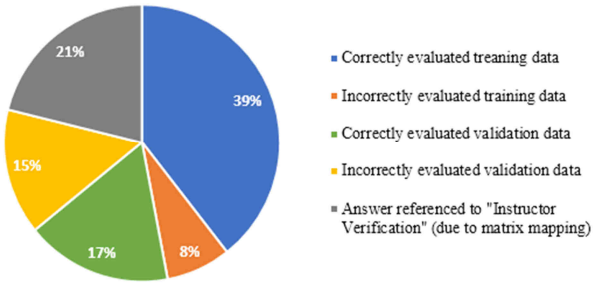

Fig. 4. The summarized results of the reduced SMS English data set and the Hungarian students' answers evaluated by the model

From the results above it can be seen, that the words in the vector space strongly affect the processable answers, because if the word is not found in the vector space, the model will be transmitted to the instructor for evaluating. Furthermore, the size of the data set strongly influences the effectiveness of the evaluation. For SMS, the larger size of dataset allowed the network to 'learn better' from the patterns, but the opposite of it with student answers, where the network did not recognize the patterns. However, the model is very sensitive to data size, but it is not to the language to be evaluated. By replacing the vector space and the dictionaries, the model is suitable for evaluating short text responses written in other languages.

4. Conclusions

The current version of the model fulfills the objective of being able to evaluate short text answers, but in the case of the Hungarian student answers, this expectation was

only partially fulfilled. The reason for this is the large amount of data, which is required for the operation of the model, which requirement does not occur in eMax systems. More detailed testing of the students' answers will take place after the data collection, as well as the comparison with the eMax system.

One of the possibilities for further development of the model is to create a vector space, which can be used not only for a certain answer, but also for a multiple answer. Consequently, it is expedient to expand the model so that it cannot only process a single question's answers, but rather process for more (e.g. questions of a subject). Another useful development of the model could be the reducing of a number of training data, which would greatly facilitate the usability of the model (usually less than 100 students answer available per question, in other word, there are a couple of ten students in a group).

References

[1] Sima D., Schmuck B., Szollősi S., Miklós Á. Intelligent short text assessment in eMax, *Proceedings of the Eight Africon Conference*, Windhoek, South Africa, 26-28 September 2007, Paper 4401593.

[2] Burrows S., Gurevych I., Stein B. The eras and trends of automatic short answer grading, *Internal Journal of Artificial Intelligence in Education*, Vol. 25, No. 1, 2015, pp. 60–117.

[3] Brassai S., Bakó L. Visual trajectory control of a mobile robot using FPGA implemented neural network, *Pollack Periodica*, Vol. 4, No. 3, 2009, pp. 129–142.

[4] Bakó L, Brassai S. Embedded neural controllers based on spiking neuron models, *Pollack Periodica*, Vol. 4, No. 3, Dec 2009, pp. 143–154.

[5] *Document-specific word2vec, Training corpus*, http://kbpedia.com/use-cases/document-specific-word2vec-training-corpuses/, (last visited 31 December 2017).

[6] Norvig P. *How to write a spelling corrector*, http://norvig.com/spell-correct.html, (last visited 31 December 2017).

[7] Liu J., Meng F., Yong Z., Liu B. Character-level neural networks for short text classification, *International Smart Cities Conference,* (ISC2), Wuxi, China, 14-17 Sept 2017, doi: 10.1109/ISC2.2017.8090812.

[8] Bruckner L., Kiss T. Introduction to Business Informatics, (in Hungarian) Akadémia Kiadó, Budapest, 2009.

[9] Huy T. N., Minh-Le N. Sentence modeling with deep neural architecture using lexicon and character attention mechanism for sentiment classification, *Proceedings of the 8th International Joint Conference on Natural Language Processing*, Taipei, Taiwan, November 27 - December 1, 2017, pp. 536–544.

[10] Huy N., Minh-Le N. A deep neural architecture for sentence-level sentiment classification in Twitter Social Networking, *Conference of the Pacific Association for Computational Linguistics*, Yangon, Myanmar, August 16 – 18, 2017, pp. 15–27.

[11] Kim Y. Convolutional neural networks for sentence classification, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing,* Doha, Qatar, 25-29 October 2014, pp. 1746–1751,

[12] Severyn A., Moschitti A. Learning to rank short text pairs with convolutional deep neural networks, *Proceedings of the 38th Internal ACM SIGIR Conference on Research and Development in Information Retrieval*, Santiago, Chile, 9-13 Aug 2015, pp. 373–382.