https://www.sworldjournal.com/index.php/swj/article/view/swj34-01-115
DOI: 10.30888/2663-5712.2025-34-01-115

UDC 004.853

NEURAL NETWORK TECHNOLOGY FOR DETECTING ERRORS IN TEXT DOCUMENTS

Udovenko S.G.
d.t.s., prof.
ORCID:https://orcid.org/0000-0001-5945-86-47
Zatkhey V.A.
c.t.s., as.prof.
ORCID:https://orcid.org/0000-0003-4426-7789
Teslenko O.V.
c.t.s., as.prof.
ORCID:https://orcid.org/0000-0003-3105-9323
Simon Kuznets Kharkiv National University of Economics,
Kharkiv, Nauky Ave. 9-A, 61165

Abstract. *The goal of this work is to develop a modified technology for error detection in text documents using a multilayer perceptron neural network.*

The proposed technology is implemented using an autoassociative neural network trained on a corpus of parallel texts containing distorted and corrected sentences. The feasibility of using the principal component method for preprocessing input text data is investigated.

The test results confirm the effectiveness of the application of the studied technology for detecting errors in polythematic text documents.

Keywords: *electronic text analysis, error detection in text documents, neural network modeling, principal component method, autoassociative three-layer perceptron*

Introduction.

An important task in the automatic analysis of electronic texts is the correction of grammatical and morphological errors (GEC) [1]. Modern search engines and text editors (Google, Word, META) partially solve this problem. They contain spelling correctors that preserve all word forms and error statistics. Spell correctors of this type work well in cloud computing, but demonstrate low speed on personal computers with limited computing resources. In this case, spell checkers, designed for spelling, generally work well only for single errors in words contained in the dictionary. Text checking in text analysis and error correction systems can be carried out in offline mode, when a protocol of comments on the text is generated, or in online mode, when errors are corrected as they are discovered (usually after receiving corresponding confirmation from the user). When an error is detected, the system can suggest a correction, and if several options are available, an ordered list of them. Comments on the text can also be of various nature. They can be local (indicating the text fragment

containing the error) or global (issuing a diagnostic message applicable to the entire text).

It should be noted that there are currently no universal systems for editing draft electronic documents. The effectiveness of existing error correction methods in natural language electronic text systems depends on the nature of the specific tasks being solved during such processing. The language and subject matter of the text, as well as the type of errors being corrected, the text presentation format, the complexity of the correction algorithm, and the quality requirements for the correction, are of great importance. One important practical task associated with the need to correct errors and typos in draft electronic texts is the analysis of foreign-language documents written by users who are not native speakers of these languages. Such documents typically contain a large number of grammatical and morphological errors and require editing. Machine learning and specialized neural network models are promising for solving problems of automatic editing of such documents [2]. Recently, neural language correction (NLC) models have been developed, showing encouraging results. Such models can be effectively used in GEC tasks, as they allow for the correction of erroneous phrases and individual grammatical errors that were not detected in the pre-editing dataset. By implementing NLC in GEC systems, the quality of grammatical error correction in source electronic texts during the pre-editing stage is significantly improved.

The goal of this work is to develop a modified technology for error detection in text documents using a multilayer perceptron neural network.

Main part.

The basic technology for detecting errors in texts is based on an encoder and decoder implemented using a recurrent neural network (RNN) trained on a corpus of parallel texts containing distorted and corrected sentences.

The distorted input sentence is processed by the correction system, after which the corrected sentence is generated at the output. Thus, the correction model consists of an encoder and decoder implemented using a pyramidal bidirectional RNN. The decoder uses the so-called "attention mechanism" to establish a connection with the encoded representation and generate the output sentence character by character.

The neural network model works on a set of symbols, as in the codec, and as in the decoder. For this purpose, the explanation is provided. First of all, it is not transmitted that the input data is checked for spelling, through which spelling errors are often narrowed in the sentences that are included in the data sets. One of the shortcomings of the "coder-decoder" architecture lies in the fact that the speech for quite a long time may be compressed into a vector of a fixed size. Another disadvantage is that the processing power is significantly reduced, since the "coder-decoder" model is small in size. Thus, it is possible to draw conclusions about the fact that for the successful processing of long speeches, the representative power of the encoder must be great, and this, as a rule, means that the model can also be great [3]. In the process of selecting the optimal model for the perfect adjustment, a large number of parameters are taken into account, which influence the implementation of the task. To eliminate the problem of the great need for information for completing tasks such as those considered, data compression is used (with or without costs). This class of problems, such as reducing the supra-dimensionality of information in the input signal, occurs with a variety of different types of neurointervention architectures.

In contrast to the traditional methods of constraint associated with the appearance of superhumanity, a neural measure in the case of a high-value task is constrained to escape from the exhaustion of insufficient resources. The topology of the network and its algorithm are such that large-dimensional data needs to be transferred from the input of the neural network to its outputs through an equally small-sized channel. To implement this kind of compression, a multi-ball perceptron of an advanced architecture can be used: a number of neurons in the input and output balls, but, however, a similar data size so that squeeze; between these balls one or more intervening balls of a smaller size are rotated. The number of intermediate balls indicates the level of foldability of data transformation. For example, a border with three intervening balls can produce better results than compression on the initial data, and can give the best result in real situations. This is due to the fact that the input data can sometimes become stale, which does not bear any connection to reality.

The input data for the measurement is formed in such a way that the outputs have

the same set of signals as the input. During the robotic process, the algorithm of the reversal expansion of the grinding minimizes the grinding. This means that the connections from the input neurons to the sphere and, approximately, to the middle sphere will work on signal compression, and the neurons will work on its decompression. In practical cases, the edge will be divided into two. The output of the first layer is transmitted through the linking channel and fed to the input of the other, which causes decompression. The key food here is the star signal. The simplest measure is the structure of direct signal transmission: signals pass from the inputs through the input elements and, in turn, arrive at the output elements. This structure leads to stable behavior. To improve the efficiency of approximation in multi-ball perceptrons (MLP), a number of enhancements are used.

First, the logistic activation function in the output ball can be replaced with a linear one, which does not change the level of activation (activation functions only change in the output ball; in intermediate balls, as before, are deprived of logistic and hyperbolic activation functions). The linear activation function cannot be saturated, and therefore it is necessary to extrapolate (in which case the logistic functions of the front levels still allow saturation at higher levels).

Linear activation functions in MLP can call out the calculation difficulties in the expansion expansion algoritm, so that with its vicoristic traces there is a small (less than 0.1) reduction in the initial speed. Descriptions are added for extrapolation purposes. Alternatively, you can change the target range of the min-max scaling function. As a result, the initial constraints will appear equal to the middle part range of output values. If this range of values is small, and its boundaries are close to the value of 0.5, then the average proportion of the sigmoid curve will be seen, which means that it is “most linear”, As a result, we will use practically the same circuit as in a different linear output ball. Such a boundary can be extrapolated in the singing boundaries, and then become saturated.

To increase the efficiency of learning by the error backpropagation algorithm in autoassociative neural networks, it is possible to use the algorithm of preliminary layer-by-layer initialization of a multilayer network, which should then be trained by the

backpropagation method. Today, there are examples of constructing nonlinear principal components using multilayer autoassociator networks [4]. One of the most effective approaches to solving the problem of data compression in order to isolate the most significant information is the factor analysis apparatus, which has found wide application in empirical data processing problems in various fields.

The main idea of factor analysis, which assumes the presence of a priori unknown hidden factors, leads to the following informal problem: observing a large number of measured parameters (indicators), identify a small number of parameter-factors that mainly determine the behavior of the measured indicators, or in other words: knowing the values of a large number of functions of the measured parameters, establish the corresponding values common to all functions of the arguments-factors and restore the form of these functions.

The input for factor analysis is an (N × n)-dimensional matrix of observations

$$X(N)=\begin{pmatrix} x_1(1) & x_2(1) & \dots & x_n(1) \\ x_1(2) & x_2(2) & \dots & x_n(2) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(k) & x_2(k) & \cdots & x_n(k) \\ \vdots & \vdots & \ddots & \vdots \\ x_1(2) & x_2(2) & \cdots & x_n(N) \end{pmatrix}=\begin{pmatrix} x^T(1) \\ x^T(2) \\ \vdots \\ x^T(k) \\ \vdots \\ x^T(N) \end{pmatrix}, \quad (1)$$

consisting of N vectors $x(k)=(x_1(k),x_2(k),...,x_n(k))^T$ and the autocorrelation $(n \times n)$ matrix $R(N)$:

$$R(N)=\frac{1}{N}\sum_{k=1}^{N}(x(k)-\overline{x}(N))(x(k)-\overline{x}(N))^T=\frac{1}{N}\sum_{k=1}^{N}\tilde{x}(k)\tilde{k}^T(k), \quad (2)$$

$$\overline{x}(k)=\frac{1}{N}\sum_{k=1}^{N}x(k), \quad (3)$$

$$\tilde{x}(k)=x(k)-\overline{x}(k). \quad (4)$$

One of the most common and effective methods for finding factors is the principal component method or component analysis, which has found wide application in data compression, pattern recognition, coding, image processing, and spectral analysis. The task of component analysis is to project data vectors from the original n-dimensional

space into the *m*-dimensional ($m<n$) space of principal components and is reduced to finding a system of $w_1, w_2, \ldots, w_m$ orthonormal eigenvectors of the matrix $R(N)$ such that $w_1 = (w_{11}, w_{12}, \ldots, w_{1n})^T$ it corresponds to the largest eigenvalue λ_1 of the matrix $R(N)$, w_2 is the second largest eigenvalue λ_2, etc. In other words, the problem is reduced to finding solutions to the matrix equation (5) such that $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda^m \geq \varepsilon \geq 0$ and $\|w_j\|=1$:

$$(R(N) - \lambda_j I_n) w_j = 0. \tag{5}$$

The dimension of the principal component space *m* is determined, as a rule, from empirical considerations and the required degree of compression of the data array. Thus, in algebraic terms, the solution of the factorial problem is closely related to the problem of eigenvalues and finding the rank of the correlation matrix; in the geometric sense, it is the problem of transition to a space of lower dimension with minimal loss of information; in the statistical sense, it is the problem of finding a set of orthonormal vectors in the input space that "take" on themselves the maximum possible variation of the data, and finally, in the algorithmic sense, it is the problem of sequentially determining the set of eigenvectors $w_1, w_2, \ldots, w_m$ by optimizing local criteria that form the following global objective function under the constraints $w_j^T w_l = 0$ $j \neq l$, $w_j^T w_j = 1$:

$$E^k = \frac{1}{k} \sum_{j=1}^{m} E_j^k = \frac{1}{k} \sum_{j=1}^{m} \sum_{p=1}^{k} (w_j^T \tilde{x}(p))^2. \tag{6}$$

The first principal component w_1 can be found by maximizing the criterion by solving a nonlinear programming problem using uncertain Lagrange multipliers:

$$E_1^k = \frac{1}{k} \sum_{p=1}^{k} (w_1^T \tilde{x}(p))^2. \tag{7}$$

However, if data processing must be carried out in real time, neural network technologies come to the fore, among which the self-learning rule should be noted, with the help of which the first principal component can be isolated:

$$\begin{cases} w_1(k+1) = w_1(k) + \eta(k)(\tilde{x}(k) - w_1(k) y_1(k)), \\ y_1(k) = w_1^T(k) \tilde{x}(k), w(0) \neq 0 \end{cases} \tag{8}$$

Next, each vector $\tilde{x}(k), k = 1, 2, \ldots, N$ is subtracted from its projection onto the first

principal component and the first principal component of the differences is calculated, which is the second principal component of the original data. The third principal component is calculated by projecting each input vector $\tilde{x}(k)$ onto the first two components, subtracting this projection from $\tilde{x}(k)$ and finding the first principal component of the differences, which is the third principal component of the original data set. The remaining principal components are calculated recursively according to the described strategy.

The problem of nonlinear factor analysis can be effectively solved using an autoassociative three-layer perceptron, known as the Bottle neck (Fig. 1).

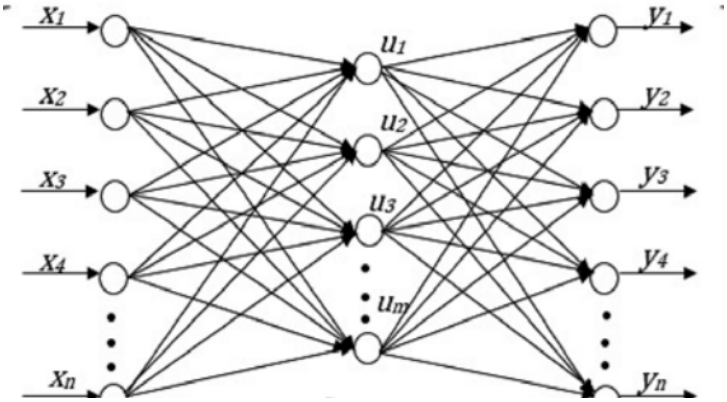

Figure 1 - Three-layer perceptron of the "Bottle neck" type

Training can be performed using any backpropagation procedure with the difference that the input signal $x(k)$ itself, which is to be compressed, is used as the training image $d(k)$. Note that the zero layer of a three-layer perceptron receives an n-dimensional vector of input signals $x(k)$ ($n_0 = n$), the first hidden layer contains $n_1 = n$ neurons, the second hidden layer $n_2 = n < n$ neurons, and the output layer $n_3 = n$. The goal of associative learning is to restore at the output of the network the signal $o^{[3]}(k)$, which best approximates the input signal $x(k)$. The actual compression of information occurs in the second hidden layer, which contains a smaller number of neurons than the first and output layers. It is from the output of the second hidden layer that the "compressed" signal $y(k) = o^{[2]}(k)$, is removed, and as a result of this approach to information compression, an optimal solution to the nonlinear factor analysis problem is achieved. We also note that if an adaptive linear associator is used as the neurons of

the “Bottle neck” perceptron, we arrive at a solution to the standard component analysis problem, which has an advantage over some multilayer perceptrons in terms of learning speed. Let us consider the procedure for training a neural network using the conjugate gradient algorithm and multiple linear regression. It has been proven that for networks of the multilayer perceptron (MLP) type, the error backpropagation learning algorithm is optimal. Classical backpropagation is a first-order gradient descent algorithm, according to which the network weights are adjusted in the anti-gradient direction. It has some disadvantages, such as linear convergence and the possibility of falling into the local minimum zone. To overcome these difficulties, the algorithm is subjected to various modifications: calculating the gradient based on the second-order derivative matrix, using fuzzy logic algorithms, simulated annealing. Neural network training is a multi-criteria nonlinear optimization problem, the goal of which is to find the optimal set of weight coefficients (function extrema) to minimize the network error. Let us consider the application of the conjugate gradient algorithm, which has greater efficiency compared to the steepest descent algorithm, and the multiple linear regression method for initializing the weight coefficients. In the neural network training problem, the conjugate gradient algorithm is used to minimize a quadratic function of the form

$$f(w) = \frac{1}{2} w^T G w - b^T w + c, \tag{8}$$

де G – Hessian matrix.

The direction of searching for the extremum is chosen so that it is orthogonal and related to all previous directions:

$$p_k = -g_k + \delta_{k-1} p_{k-1}, \tag{9}$$

where g_k is the actual value of the gradient direction.

The change in weights is calculated by the formula:

$$w_{k+1} = w_k + \alpha_k p_k. \tag{10}$$

The coefficients α and β are used to determine the conjugate direction. The value α_k must minimize $f(w_{k+1})$, for which the golden section method or another linear minimum search algorithm can be applied. The coefficient δ_k is calculated using the

Fletcher-Reeves formula:

$$\beta_k = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}. \quad (11)$$

For quadratic functions, the conjugate gradient method finds the minimum in n steps, so the algorithm needs to restart every $n + 1$ steps. For the Fletcher-Reeves method, this is done using δ_k. To calculate the gradients, this method uses the neural network error objective function:

$$f(w) = \frac{1}{2N} \sum_n \sum_j (y_{nj} - d_{nj})^2, \quad (12)$$

where N is the number of training images; y_{nj} and d_{nj} are the actual and desired outputs of the network, respectively.

The choice of the starting point, which is provided by the initial initialization of the network weights, plays an important role in the learning process. Initialization with random uniformly distributed values, which is typical for gradient algorithms, does not guarantee the avoidance of local minima during the descent process. This stage can be implemented using multiple linear regression (MLR). In this case, the weights of the connections of the input and hidden layers are still initialized randomly, and the weights of the connections of the hidden and output layers are calculated using MLR.

The experiment used a dictionary that included 98 characters: a set of ASCII print characters, special characters <sos>, <eos> and <unk>, which denote the beginning, end of a sentence and unknown characters, respectively. The experiment used a data set that was obtained from the set of common tasks CoNLL 2013 and 2014 and which contains about 60 thousand sentences from essays written by users who learn English with corrections and descriptions of error types. The same initialization of weight matrices in the range [-0.1, 0.1] and with zero initialization of shifts were used. In this case, a test set was used that contains 100,000 and 1,000 records from approximately 550,000 and 5,500 parallel sentences, respectively. 5,500 sentences from the training set were also selected to be used as a separate development set for model and parameter selection. To train the network, the CoNLL 2014 test data with error-containing sentences was included in the experimental program. The data augmentation procedure

creates synthetic errors for the two most common types of errors in the development set: errors in articles or demonstrative pronouns (ArtOrDet) and errors in noun number (Nn). For ArtOrDet errors, the probability that the article or pronoun was removed, replaced with another demonstrative pronoun, or inserted before the beginning of a noun phrase was estimated. For Nn errors, the probability that nouns were replaced was estimated. The Standford CoreNLP Toolkit was used to obtain sentence parsing. Language model datasets can significantly improve performance. Expanding the language model weight λ typically improves recall at the expense of accuracy. On the other hand, using edited classification to filter out erroneous edits improves accuracy, often with less impact on recall. During the experiment, promising improvements were obtained by expanding the data, which allowed to increase the F0.5 indicator on the basis of the development set (Fig. 2). For the two types of errors where data were synthesized (article or pronoun and noun number), an increase in memory recovery was observed (Table 1).

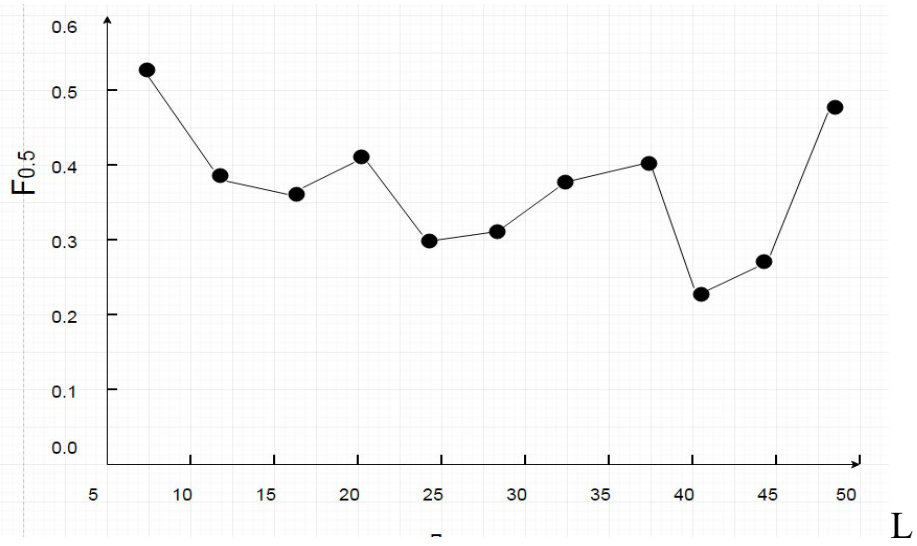

Figure 2 - Dependence of F-scores on the length of the input sentence (L)

Figure 3 shows a form with an example of sentence processing: the window of this form displays a proposed solution to improve the quality of the sentence, as well as highlighted replaced phrases or words.

It should be noted that the network that was used was previously trained taking into account the tasks of correcting certain types of errors.

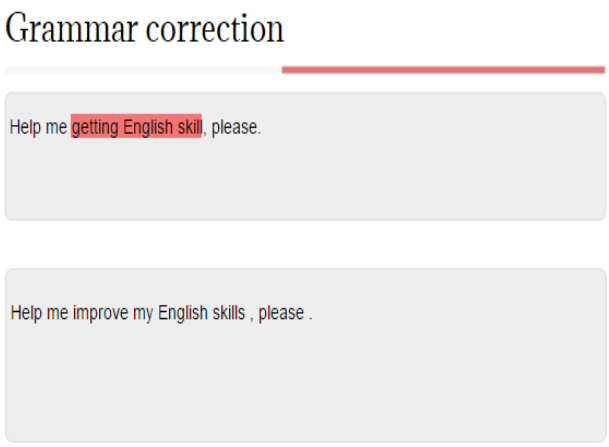

Figure 3 - An example of a screen form with the correction result

Table 1 - Network performance metrics

Type	Quantity	R with extension	R without extension
ArtOrDet	727	21,08	29,27
Wci	441	2,35	1,59
Nn	405	31,59	50,09
Prepositions	325	13,11	8,13
Word forms	228	27,09	19,03

The results of the experiment indicate an acceptable quality of correction of grammatical and morphological errors in text documents when using neural network technology using an associative three-layer perceptron. In particular, out of 365 errors and typos related to the absence, insertion and replacement of letters in the analyzed tests, 343 errors were detected and corrected.

Conclusions.

The paper presents the results of a study of a modified technology for detecting errors in text documents using a multilayer perceptron neural network.

The possibility of increasing the efficiency of correcting errors in text documents using an autoassociative three-layer perceptron is shown. The choice of a neural network model that operates at the symbol level is due to the fact that standard neural models of machine translation are not always effective for processing polythematic documents containing multi-digit numbers, special characters, web addresses, etc.

Practical results are presented with examples that confirm the prospects for using the technology under consideration for detecting and correcting errors in English-language text documents.

References:

1. Susanto R., Phandi P. System combination for grammatical error correction / In Proc. of EMNLP, 2014. – P. 21–33.

2. Chala L., Udovenko S., Hrynev S. Method of neural network correction of errors in editable electronic texts. Bionics of intelligence, 2017. – Issue 1 (88). – P. 15-21.

3. Zhang F. Design and application of an automatic scoring system for english composition based on artificial intelligence Technology. Int. J. Adv. Comput. Sci. , 2023. 14(8). – P. 195–205.

4. Kalchbrenner, N., Blunsom P. Recurrent Continuous Translation Models. In Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2013, P. 1700 – 1709.

Анотація. *Метою цієї роботи є розробка модифікованої технології виявлення помилок у текстових документах за допомогою багатошарової персептронної нейронної мережі. Запропонована технологія реалізована за допомогою автоасоціативної нейронної мережі, навченої на корпусі паралельних текстів, що містять спотворені та виправлені речення. Досліджено доцільність використання методу головних компонент для попередньої обробки вхідних текстових даних.*

Результати тестування підтверджують ефективність застосування досліджуваної технології для виявлення помилок у політематичних текстових документах.

Ключові слова: *аналіз електронного тексту, виявлення помилок у текстових документах, моделювання нейронних мереж, метод головних компонент, автоасоціативний тришаровий персептрон.*

Статтю надіслано: 25.11.2025 г.