Technical sciences

UDC 681.324

Yershov Oleksandr
Student of the
National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»
Єршов Олександр Ігорович
студент
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Ершов Александр Игоревич
студент
Национальный технический университет Украины
«Киевский политехнический институт имени Игоря Сикорского»

Kysliak Serhii
Senior Lecturer of the
National Technical University of Ukraine
«Igor Sikorsky Kyiv Polytechnic Institute»
Кисляк Сергій Володимирович
старший викладач
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Кисляк Сергей Владимирович
старший преподаватель
Национальный технический университет Украины
«Киевский политехнический институт имени Игоря Сикорского»

PROTEIN SEQUENCES CLASSIFICATION BY CONVOLUTION NEURAL NETWORK

КЛАСИФІКАЦІЯ БІЛКОВИХ ПОСЛІДОВНОСТЕЙ ЗА ДОПОМОГОЮ ЗГОРТКОВОЇ НЕЙРОННОЇ МЕРЕЖІ

КЛАСИФИКАЦИЯ БЕЛКОВЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ С ПОМОЩЬЮ СВЕРТОЧНОЙ НЕЙРОННОЙ СЕТИ

Summary. *Classification of protein sequences by machine learning methods requires a lot of time and computing power. The key to successful data classification for machine learning is choose the best algorithm. Neural networks are able to study large amounts of data in a short period of time without requiring significant processing of input information. Convolution Neural Networks (CNN) are usually used in Computer Vision, but recently they allow you to get good results for various tasks on Nature Language Processing (NLP).*

Key words: *classification, deep learning, oxidoreductase, transferase, hydrolase, convolutional neural network, nature language processing.*

Анотація. *Класифікація білкових послідовностей методами машинного навчання потребує великих витрат часу та обчислювальних потужностей комп’ютера. Вибір найкращого алгоритму для машинного навчання є запорукою успішної класифікації даних. Нейронні мережі здатні вивчати великі об’єми даних за короткий проміжок часу, при цьому не потребуючи значної обробки вхідної інформації. Згорткові нейронні мережі(CNN-Convolution Neural Network) зазвичай використовують в Computer Vision, проте останнім часом вони дозволяють отримати не погані результати для вирішення різних задач, що пов`язані з обробкою тексту (NLP-Nature Language Processing).*

Ключові слова: *класифікація, глибоке навчання, oxidoreductase, transferase, hydrolase, згорткова нейронна мережа, класифікація текстових даних.*

Аннотация. *Классификация белковых последовательностей методами машинного обучения требует больших затрат времени и вычислительных мощностей компьютера. Выбор лучшего алгоритма для машинного обучения является залогом успешной классификации данных. Нейронные сети способны изучать большие объемы данных за короткий промежуток времени, при этом не требуя значительного обработки входящей информации. Сверточные нейронные сети (CNN-Convolution Neural Network) обычно используют в Computer Vision, однако в последнее время они позволяют получить неплохие результаты для решения различных задач, которые связаны с обработкой текста (NLP-Nature Language Processing).*

Ключевые слова: *классификация, глубокое обучения, oxidoreductase, transferase, hydrolase, сверточная нейронная сеть, классификация текстовых данных.*

Introduction. Advances in biological and medical technologies have contributed to the emergence of vast amounts of biological data, such as medical images and protein sequences [1]. In the age of big data, the transformation of biomedical data into valuable knowledge is one of the most important tasks of bioinformatics [2]. Biological data are often complex, heterogeneous and difficult to interpret, so they are a good example for deep learning methods [3]. Deep learning is a new field of machine learning research, the aim of which is to bring machine learning closer to one of its original goals - artificial intelligence. In recent years, deep neural networks as machine learning tools have become increasingly popular. The availability of large computing resources, large amounts of data, new algorithms for learning deep models and easy-to-use libraries for learning neural networks are the drivers of development in this area [4].

The constant increase in biological information in all biomedical fields underscores the potential for an even greater role that deep learning can play in future research. The presence of a large amount of data requires the improvement of standard methods of bioinformatics, which can be used for the annotation of proteins, analysis of gene expression, identification of protein families, and others.

Convolution Neural Networks are neural networks that used mainly for computer image classification, but studies [6,7,8] have shown that this deep learning algorithm is able to demonstrate extremely accurate prediction for NLP tasks - classification of text data.

Neural networks are able to solve such a problem as binary classification of protein sequences. According to [9; 10], neural networks can simultaneously classify a large number of classes, which indicates the flexibility and versatility of neural networks for a large number of tasks. However, it should be noted that neural networks are characterized by the problem of "overfitting", which leads to "learning" of training data by the network and does not allow to obtain accurate forecast results on real data.

Thus, by creating a convolutional neural network architecture to classify three groups of protein sequences: oxidoreductase, transferase and hydrolase, regulating different components of the neural network, with the addition of tools that prevent "overfitting", we obtain an optimal convolutional neural network model that can classify unlimited protein classes quickly and efficiently.

In the learning process, the indicators are interpreted in modern visualization using a graph of data movement, which allows you to control the work and understand the movement of data within the network.

Materials and Methods. In this study, the convolutional neural network architecture was built in the Python programming language, using the Google Colaboratory (Colab) service with access to a graphics processor (GPU) and an open Tensorflow platform for machine and deep learning. The text input data of the neural network was subjected to the method of tokenization - the transformation of a text sequence into a vector sequence of integers [11].

The convolutional neural network has an architecture (Fig.1):

- embedding layer, which is initialized by random scales and studies the embedding - the position of the word in the vector space, for all words in the training data set [12]. This layer adapts the model to a specific dataset.
- convolutional layers, which are the main building block of CNN. The parameters of the layer consist of a set of filters for training [13]. During the direct pass, each filter performs a convolution of the input dimension, calculating the scalar product of the filter and input data, and forming an excitation map of this filter. As a result, the network learns to activate filters when it detects a specific type of feature in a specific spatial position at the input.
- max-pooling layer, the function of which is to gradually reduce the spatial dimensions of the representation to reduce the number of parameters and calculations in the network, and hence to control the dataset. The maximization aggregation layer works independently on each fragment of the input depth and changes it spatially, using the operation "MAX" (selection of the maximum value) [14].
- flatten layer, on which the result obtained on the previous layers is aligned in the combined map, presented in the form of a column, for presentation in the next layer [15].

- fully-connected or dense layers, which are divided into a first fully-connected layer and a fully-connected output layer [16]. The first of them takes the input data of the performance analysis and uses the scales to predict the correct class. The second calculates the finite probabilities for each class.

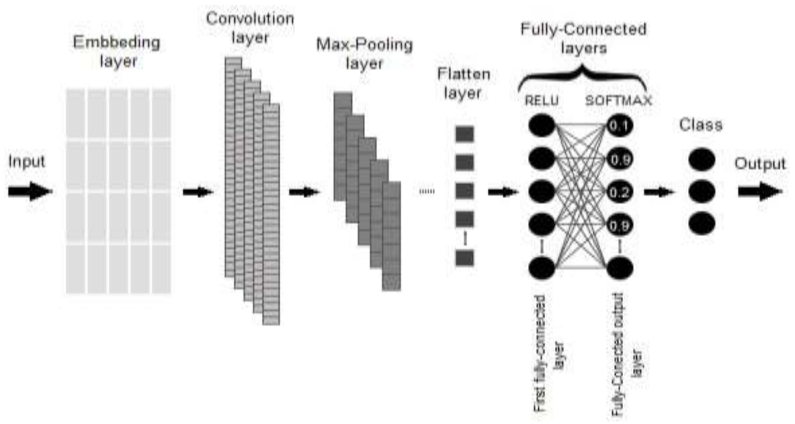


Fig. 1. Convolutional neural network architecture for protein sequence classification

It is necessary to indicate the stages of CNN training:

1. Initialization of initial weights for all neurons;
2. "Forward propagation" - moving the entire training dataset on the neural network and get the result;
3. The loss function captures the difference between the correct classification result and the actual output of the model, taking into account the current weight of the model, thereby indicating how close the model from the correct result [17]; 4. "Backpropagation" - to minimize the error function [18];
4. Weight update - weight change according to the backpropagation;
5. Iteration to convergence - the number of iterations required for convergence.

The fully-connected layers of CNN go through their own "backpropagation" process to determine the most accurate weights. The "backpropagation" algorithm performs a highly efficient search for optimal weight values using the gradient descent technique [19]. This allows you to minimize error functions with low computing resources and create a conclusion - the decision on classification.

For this problem of classification of three groups of proteins, categorical cross entropy was used as a function of error [20]. This cost function is intended to classify only one label, ie provided that there is only one class for one protein sequence.

In order to improve the learning process of the neural network and avoid the problem of overfitting, regularization methods such as "Dropout" and "Early Stopping" were used for some models [21; 22]. The first method uses a technique where randomly selected neurons are ignored during training. They "drop-out" randomly. This means that their contribution to the activation of descending neurons is temporarily removed for the period of forward propagation and any weight updates are not applied to the neuron in the backpropagation. Accordingly, neurons will randomly "fall out" of the network during training, and other neurons will have to process the information needed to predict missing neurons. It is believed that this leads to the network learning a lot of independent internal representations. As a result, the network becomes less sensitive to the specific weights of neurons, which leads to better generalization and less chance of overfitting training data.

The "Early Stopping" method is based on the early cessation of neural network learning due to the cessation of efficiency gains. As an indicator of efficiency in this case, the error function was used, which should be minimized. Therefore, if the error function begins to increase rather than decrease as needed, then this method stops the learning process.

Metrics such as: "precision", "recall", "f1-score" were used to assess the accuracy of the forecast of each class. To assess the quality of the model as a whole - "accuracy" - the ratio of correctly predicted observations to total observations, "precision" - is the ratio of correctly predicted positive observations to total predicted positive observations, "recall" - the ratio of correctly predicted positive observations to all observations in the actual class, "f1-score" - the weighted average of the metrics "precision" and "recall".

Biological data for network training and testing were obtained from the Structural Bioinformatics Research Laboratory (RCSB) of the Protein Data Bank (PDB).

The following libraries were used for data processing and visualization: "Pandas" (http://pandas.pydata.org/) [23], "NumPy" (https://numpy.org/) [24], "Matplotlib" (https://matplotlib.org/) [25]; to create the architecture of the neural network "Keras" [https://keras.io/) [26]; to visualize the operation of the neural network "Tensorflow" (https://www.tensorflow.org/) [27] in Python programming language.

Results. Three models of the convolutional neural network were created, among which the architecture was compared and the optimal one was selected. The architecture of the first CNN model was used as a basis for creating others. To obtain the optimal model, should be determined experimentally the depth of the network and using regularization methods.

The first model contains an embedding layer, two convolution layers, followed by a max-pooling layer, a flatten layer, and two fully-connection layers. The training of this model took place in 15 predetermined epochs (Fig. 2). With each epoch, the accuracy of the forecast increases (Fig. 3), and the error function is minimized (Fig. 4). The accuracy of the prediction and the value of the error function after 15 epochs on the training and test data is shown in Fig. 5. The values of forecast accuracy metrics for each class are obtained (Fig. 6). In the process of learning the first model of CNN, regularization methods were not used.

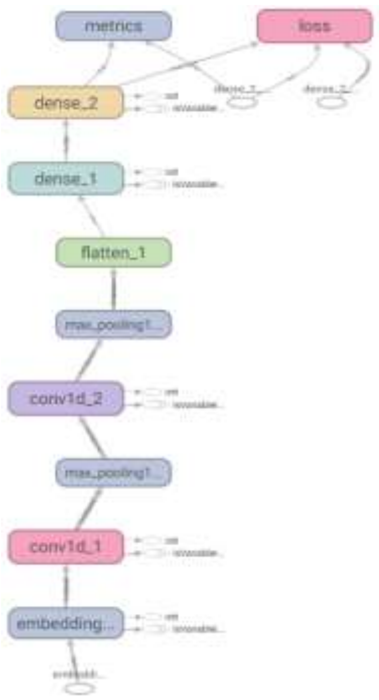


Fig. 2. Graph of data transfer during training of the first CNN model

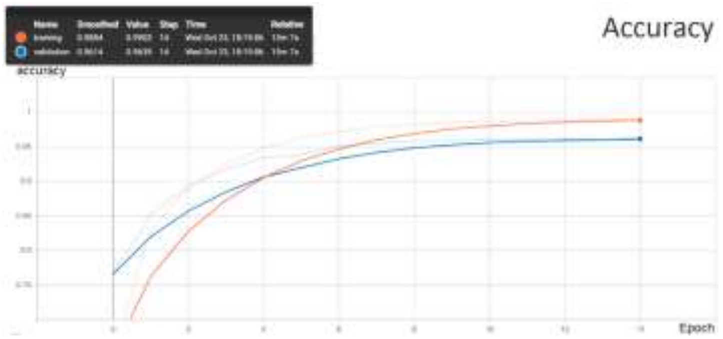


Fig. 3. Graph the increasing accuracy with each training epoch for the first CNN model

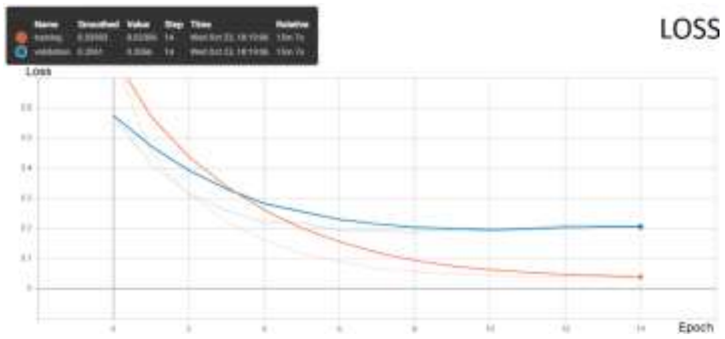
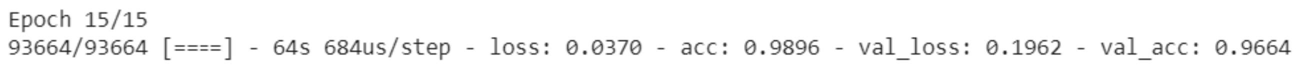
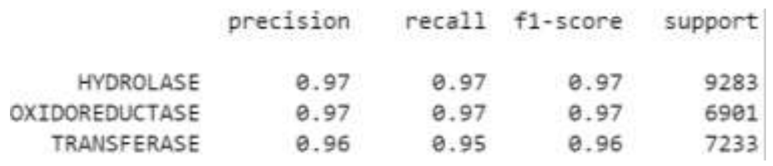


Fig. 4. Graph the minimization of the error function with each training epoch for the first CNN model



```
Epoch 15/15
93664/93664 [====] - 64s 684us/step - loss: 0.0370 - acc: 0.9896 - val_loss: 0.1962 - val_acc: 0.9664
```

Fig. 5. Accuracy values and error functions after the 15th training era for the first CNN model



```
               precision    recall  f1-score   support

    HYDROLASE       0.97      0.97      0.97      9283
OXIDOREDUCTASE      0.97      0.97      0.97      6901
  TRANSFERASE       0.96      0.95      0.96      7233
```

Fig. 6. The values of the accuracy metrics of each class for the first CNN model

The second model contains an embedding layer, three convolution layers, followed by max-pooling layer, an flatten layer and two fully-connection layers (Fig. 7). This model was trained using the "Early Stopping" method, which is why a large number of epochs were previously set. With each epoch, the accuracy of the forecast increases (Fig.8), and the error function is minimized (Fig.9). The early stop worked after the 19th era. The accuracy of the forecast and the value of the error function after 19 epochs in the training and test data are shown in Fig. 10. The values of forecast accuracy metrics for each class are obtained (Fig.11).

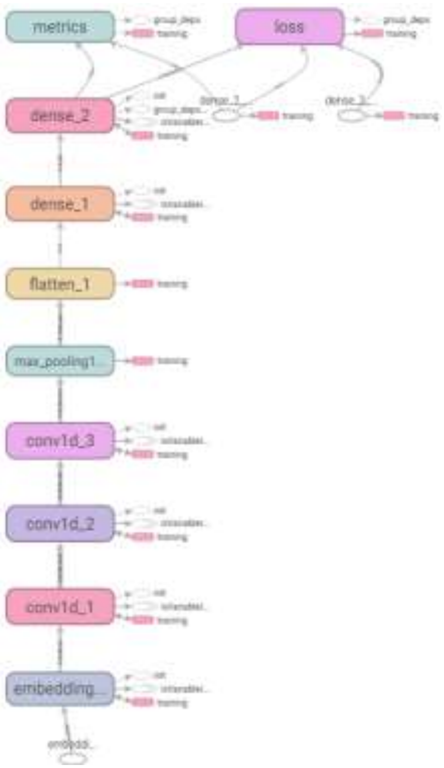


Fig. 7. Graph of data transfer during training of the second CNN model

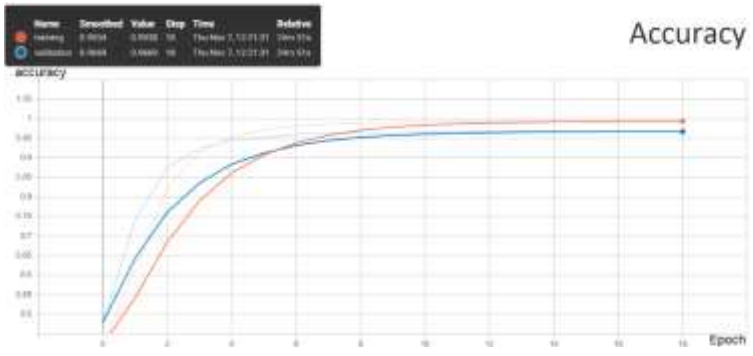


Fig. 8. Graph the increasing accuracy with each training epoch for the second CNN model

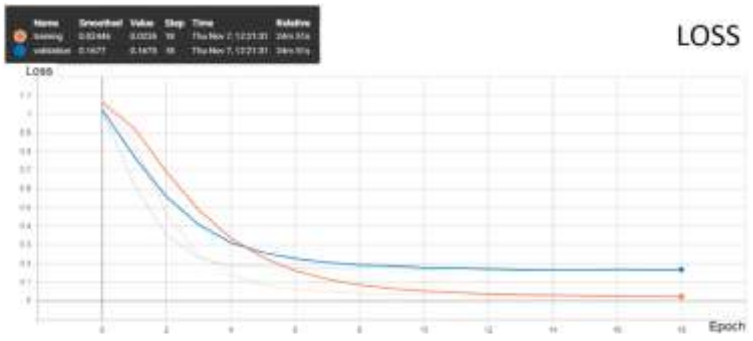
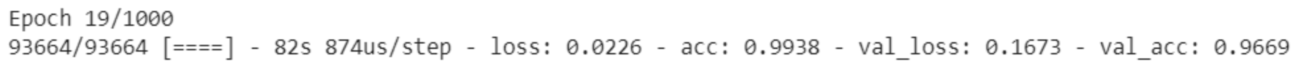
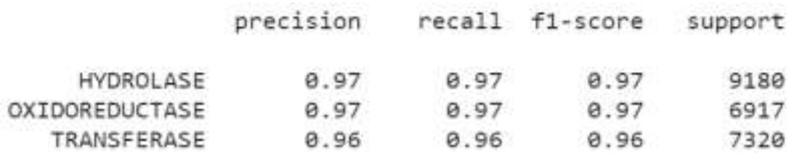


Fig. 9. Graph the minimization of the error function with each training epoch for the second CNN model



```
Epoch 19/1000
93664/93664 [====] - 82s 874us/step - loss: 0.0226 - acc: 0.9938 - val_loss: 0.1673 - val_acc: 0.9669
```

Fig. 10. Accuracy values and error functions after the 19th training era for the second CNN model



```
                 precision    recall  f1-score   support

      HYDROLASE       0.97      0.97      0.97      9180
 OXIDOREDUCTASE       0.97      0.97      0.97      6917
    TRANSFERASE       0.96      0.96      0.96      7320
```

Fig. 11. The values of the accuracy metrics of each class for the second CNN model

The third model contains an embedding layer, three convolution layers, followed by max-pooling layer, a flatten layer and two fully-connection layers (Fig.12). This model was trained using the "Early Stopping" method and the "Dropout" method, which was added twice: between the max-pooling layer and the flatten layer, between the first fully-connection layer and the output fully-connection layers. The "Dropout" method was set to "0.25" - which means the exclusion of 25% of neurons. Due to the use of regularization methods, a large number of epochs have been set beforehand. With each epoch, the accuracy of the forecast increases (Fig.13), and the error function is minimized (Fig.14). The early stop worked after 31 epochs. The accuracy of the forecast and the value of the error function after 31 epochs on the training and test data are shown in Fig.15. The values of forecast accuracy metrics for each class are obtained (Fig.16).

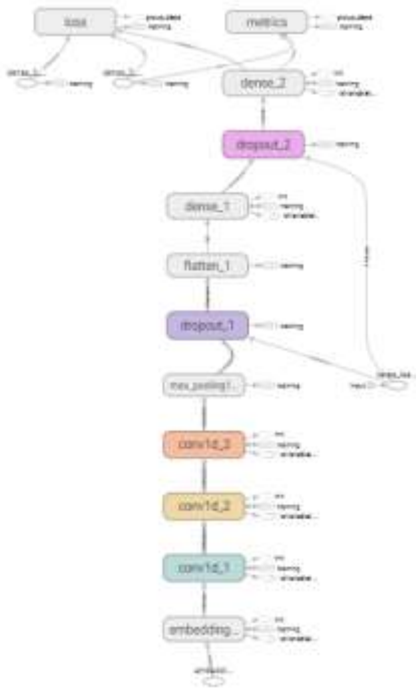
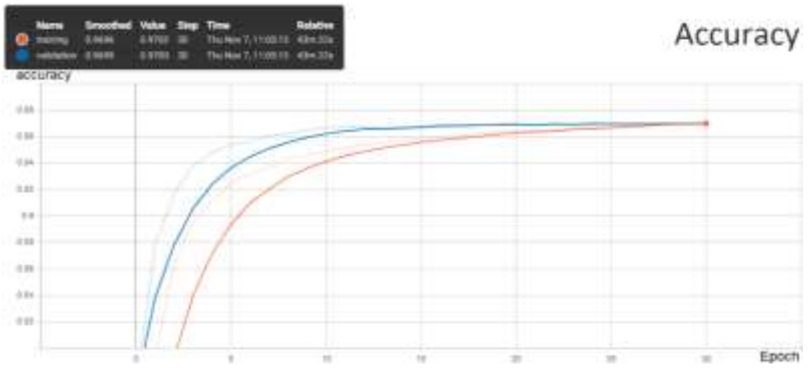


Fig. 12. Graph of data transfer during training of the third CNN model



	Name	Smoothed	Value	Step	Time	Relative
🟠	training	[illegible]	[illegible]	[illegible]	Thu Nov 7, [illegible]	[illegible]
🔵	validation	[illegible]	[illegible]	[illegible]	Thu Nov 7, [illegible]	[illegible]

Fig. 13. Graph the increasing accuracy with each training epoch for the third CNN model

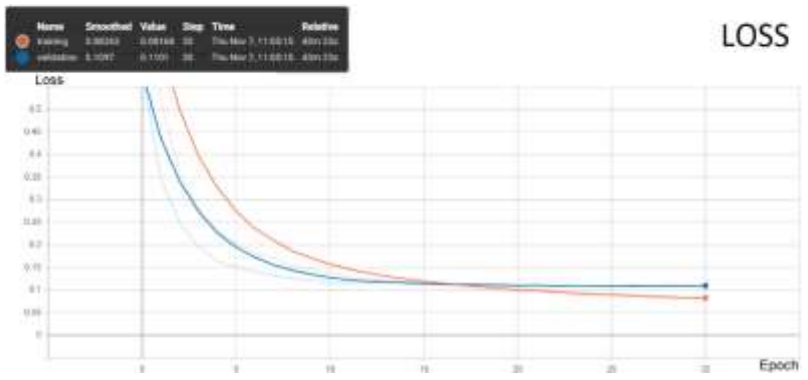
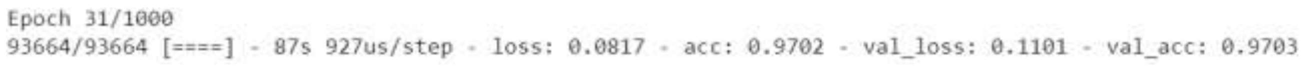
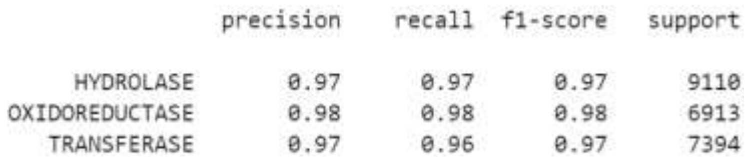


Fig. 14. Graph the minimization of the error function with each training epoch for the third CNN model



```
Epoch 31/1000
93664/93664 [====] - 87s 927us/step - loss: 0.0817 - acc: 0.9702 - val_loss: 0.1101 - val_acc: 0.9703
```

Fig. 15. Accuracy values and error functions after the 31th training era for the third CNN model



	precision	recall	f1-score	support
HYDROLASE	0.97	0.97	0.97	9110
OXIDOREDUCTASE	0.98	0.98	0.98	6913
TRANSFERASE	0.97	0.96	0.97	7394

Fig. 16. The values of the accuracy metrics of each class for the third CNN model

Discussion. According to the metrics of accuracy "precision", "recall", "f1-score" from Fig.6,11,16, calculated for each class as follows:

$$precision = \frac{TP}{TP + FP} \text{ [28],}$$

where *TP* is a number of true positive values; *FP* – is a number of false positives values;

$$recall = \frac{TP}{TP + FN} \text{ [28],}$$

where TP is a number of true positive values; FN is a number of false negative values;

$$f1\text{-}score = 2*\left(\frac{precision*recall}{precision+recall}\right)$$ [29],

where precision – value of metric "precision"; recall – value of metric "recall";

it can be argued that all three networks classify hydrolase with an accuracy of 97%. The first and second neural network models also classify oxidoreductase with an accuracy of 97%, however, the third CNN model classifies this group with an accuracy of 98%. According to the metrics "precision" and "f1-score", the first and second network models classify transferase with an accuracy of 96%, the third - with an accuracy of 97%. According to the "recall" metric, the first network model classifies transferase with an accuracy of 95%, the second and third - with an accuracy of 96%.

The overall accuracy for the first CNN on the test data is 96.64%, the second - 96.69%, the third - 97.03% (Fig.5,10,15)

The error function, in the form of a categorical cross entropy, calculated by the formula:

$$CE= -\frac{1}{N}\sum_i y_i \log(y`_i)$$ [29]

where $y`_i$–result of the last layer with softmax, forecast; y_i is the true value;

indicates the effectiveness of training (Fig. 4,9,14). For the first CNN on the test data takes the value of 0.1962, for the second - 0.1673 and for the third - 0.1101.

According to Fig.2,7,12 you can see that the training time of the first network is 15 minutes, the second - 25 minutes, the third - 43 minutes.

Conclusions. After analyzing the three architectures of the convolutional neural network, the following conclusions can be made: the first model of CNN training the fastest, however, quality indicators indicate low accuracy of classification of transferase compared to other architectures. The second CNN model has better accuracy, but compared to the third architecture, it is clear that the latter has the best accuracy and the lowest values of the loss function, which is a very important indicator of training. Given the architecture of each CNN and the quality results obtained, it can be argued that regularization methods played an important role in the learning process. Since the first model did not have any method, the second model had one method of regularization - "Early Stopping", and the third - two methods of regularization "Early Stopping" and "Dropout", which significantly affected the training.

It has become possible to increase the depth of the neural network, which means that the network better understands the data and searches for complex relationships, avoiding the problem of overfitting.

The best convolutional neural network is the third model, which contains an embedded layer, three convolution layers, a max-pooling layer, a flatten layer, and two fully-connected layers. This CNN was trained using the "Early Stopping" method and the "Dropout" method, which was added twice. CNN has an overall accuracy of 97.03% on test data, the cost value reaches 0.1101. The network classifies hydrolase, oxidoreductase and transferase with 97%, 98% and 96.6% accuracy, respectively.

References

1. Cao C, Liu F, Tan H, Song D, Shu W, Li W et al. Deep Learning and Its Applications in Biomedicine. Genomics, Proteomics & Bioinformatics. 2018;16(1):17-32.
2. Min S, Lee B, Yoon S. Deep learning in bioinformatics. Briefings in Bioinformatics. 2016;:bbw068.

3. Fioravanti D, Giarratano Y, Maggio V, Agostinelli C, Chierici M, Jurman G et al. Phylogenetic convolutional neural networks in metagenomics. BMC Bioinformatics. 2018;19(S2).
4. Jurtz V, Johansen A, Nielsen M, Almagro Armenteros J, Nielsen H, Sønderby C et al. An introduction to deep learning on biological sequence data: examples and solutions. Bioinformatics. 2017;33(22):3685-3690.
5. Ching T, Himmelstein D, Beaulieu-Jones B, Kalinin A, Do B, Way G et al. Opportunities and obstacles for deep learning in biology and medicine. Journal of The Royal Society Interface. 2018;15(141):20170387.
6. Zheng D, Pang G, Liu B, Chen L, Yang J. Learning transferable deep convolutional neural networks for the classification of bacterial virulence factors. Bioinformatics. 2020.
7. Taju S, Nguyen T, Le N, Kusuma R, Ou Y. DeepEfflux: a 2D convolutional neural network model for identifying families of efflux proteins in transporters. Bioinformatics. 2018;34(18):3111-3117.
8. Budach S, Marsico A. pysster: classification of biological sequences by learning sequence and structure motifs with convolutional neural networks. Bioinformatics. 2018;34(17):3035-3037.
9. Hanson J, Paliwal K, Litfin T, Yang Y, Zhou Y. Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks. Bioinformatics. 2018;34(23):4039–4045.
10. Szalkai B, Grolmusz V. SECLAF: a webserver and deep neural network design tool for hierarchical biological sequence classification. Bioinformatics. 2018;34(14):2487-2489.
11. Giorgi J, Bader G. Transfer learning for biomedical named entity recognition with neural networks. Bioinformatics. 2018;34(23):4087-4094.

12. Bazzan A, Engel P, Schroeder L, da Silva S. Automated annotation of keywords for proteins related to mycoplasmataceae using machine learning techniques. Bioinformatics. 2002;18(Suppl 2):S35-S43.
13. Luo X, Tu X, Ding Y, Gao G, Deng M. Expectation pooling: an effective and interpretable pooling method for predicting DNA–protein binding. Bioinformatics. 2019;36(5):1405–1412.
14. Kang Q, Meng J, Cui J, Luan Y, Chen M. PmliPred: a method based on hybrid model and fuzzy decision for plant miRNA–lncRNA interaction prediction. Bioinformatics. 2020;36(10):2986-2992.
15. Min X, Zeng W, Chen N, Chen T, Jiang R. Chromatin accessibility prediction via convolutional long short-term memory networks with k-mer embedding. Bioinformatics. 2017;33(14):i92-i101.
16. Kang Q, Meng J, Cui J, Luan Y, Chen M. PmliPred: a method based on hybrid model and fuzzy decision for plant miRNA–lncRNA interaction prediction. Bioinformatics. 2020;36(10):2986-2992.
17. Pagel K, Pejaver V, Lin G, Nam H, Mort M, Cooper D et al. When loss-of-function is loss of function: assessing mutational signatures and impact of loss-of-function genetic variants. Bioinformatics. 2017;33(14):i389-i398.
18. Parca L, Ariano B, Cabibbo A, Paoletti M, Tamburrini A, Palmeri A et al. Kinome-wide identification of phosphorylation networks in eukaryotic proteomes. Bioinformatics. 2018;35(3):372-379.
19. Zhang W, Ma J, Ideker T. Classifying tumors by supervised network propagation. Bioinformatics. 2018;34(13):i484-i493.
20. Strodthoff N, Wagner P, Wenzel M, Samek W. UDSMProt: universal deep sequence models for protein classification. Bioinformatics. 2020;36(8):2401-2409.

21. Adhikari B. DEEPCON: protein contact prediction using dilated convolutional neural networks with dropout. Bioinformatics. 2019;36(2):470–477.
22. Fu W, Dougherty E, Mallick B, Carroll R. How many samples are needed to build a classifier: a general sequential approach. Bioinformatics. 2004;21(1):63-70.
23. Abdennur N, Mirny L. Cooler: scalable storage for Hi-C data and other genomically labeled arrays. Bioinformatics. 2019;36(1):311–316.
24. Yin P, Voight B. MeRP: a high-throughput pipeline for Mendelian randomization analysis. Bioinformatics. 2014;31(6):957-959.
25. Tareen A, Kinney J. Logomaker: beautiful sequence logos in Python. Bioinformatics. 2019;36(7):2272-2274.
26. Kalkatawi M, Magana-Mora A, Jankovic B, Bajic V. DeepGSR: an optimized deep-learning structure for the recognition of genomic signals and regions. Bioinformatics. 2018;35(7):1125-1132.
27. Pagès G, Charmettant B, Grudinin S. Protein model quality assessment using 3D oriented convolutional neural networks. Bioinformatics. 2019;35(18):3313-3319.
28. Ji Y, Yu C, Zhang H. contamDE-lm: linear model-based differential gene expression analysis using next-generation RNA-seq data from contaminated tumor samples. Bioinformatics. 2020;36(8):2492-2499.
29. Xing H, Kembel S, Makarenkov V. Transfer index, NetUniFrac and some useful shortest path-based distances for community analysis in sequence similarity networks. Bioinformatics. 2020;36(9):2740-2749.