DOI 10.52209/1609-1825_2023_2_335 UDC 004.032.26

Using a Convolutional Neural Network to Convert Black-and-White Photos into a Colorized Format in the Telegram Bot

[1]**MUTOVINA Natalya,** *Cand. of Tech. Sci., Associate Professor, mutovina_natalya@mail.ru,*
[1]***ALINA Gaukhar,** *Master, Teacher, diamond_gaxa@mail.ru,*
[2]**YUCHSHENKO Olessya,** *PhD, Cand. of Tech. Sci., Associate Professor, Head of EPD, olessyayuchenko@hotmail.com,*
[1]**YANKE Nikita,** *Student, diamond_gaxa@mail.ru,*
[1]*NPJSC «Abylkas Saginov Karaganda Technical University», Kazakhstan, Karaganda, N. Nazarbayev Avenue, 56,*
[2]*Voronezh Institute of High Technologies, Russia, Voronezh, Lenin Street, 73a,*
**corresponding author.*

Abstract. *Artificial neural networks will be used to solve our problem, since they are best suited for making predictions, due to their ability to detect non-obvious connections and the most general patterns of input and output data. A diagram of the architecture of the created neural network has been developed. Lab color coding method considered. The program is implemented in the high-level language Python. To create a neural network, an open software library for machine learning TensorFlow was used. The ReLU function (linear rectifier or semi-linear element) was chosen as the activation function for all layers, except for the output one. A telegram bot has been developed, the main purpose of which is to convert the sent black-and-white images into a color format. With the help of the developed software algorithm, it is also possible to solve the problem of low quality and low resolution of images by processing them with an online neural network. After the received images can be saved.*

Keywords: *machine learning, bot, output, Python, neural networks, color, message, chat.*

Introduction

Machine learning is a class of artificial intelligence methods that are based on learning through the use of algorithms for solving similar problems. Some fields of mathematics are used to construct such methods as mathematical statistics, numerical methods, mathematical analysis, probability theory, graph theory, and various techniques for working with digital data.

Artificial neural networks will be used to solve our problem, since they are best suited for making predictions, due to their ability to detect non-obvious connections and the most general patterns of input and output data.

Having studied the functionality of messengers and made an overview of modern developments of chatbots in messengers, for example, such as: Colorizerbot or AIMagic telegram bots, with which you can improve the quality of photos and turn an archived black-and-white photo into a color one. Taking into account the experience of development in this direction, it allowed to avoid a number of errors when processing photos, for example, it turned out that Telegram reduces the quality of the original images, so you need to send the original image (without loss of quality).

The final product will be a telegram bot that converts sent black-and-white images into a color format. Also, the bot will send a message to the user with information about itself and an explanation about how to use it, after typing the command '/start'. The bot will not have any other functionality.

The Python programming language was chosen as the main tool, since a huge number of libraries and frameworks for creating and working with artificial intelligence were developed for this specific language. TensorFlow framework will be used to create a neural network because of its rich toolkit and high-level interface. The application that will provide access to the capabilities of the neural network will be a telegram bot. The python-telegram-bot library will be used to create he bot.

Research methods

A convolutional neural network was chosen as the neural network architecture, since such neural networks are able to find the most general patterns in images, which is perfect for colorizing an image.

Figure 1 shows the architecture diagram of the created neural network. A two-dimensional matrix of the pixel values of a black-and-white image is fed to the input of the neural network. At the output, the

neural network shows a three-dimensional matrix, where the number of rows and columns is the same as the input one, but its height is already 2, the latter matrix is the pixel values of 'a' and 'b' channels of the Lab format.

Lab is a color coding method consisting of 3 parts: 'L', 'a' and 'b'. 'L' is the black and white layer of the image, the values of 'L' vary from 0 to 100; 'a' is the color from green to red, 'b' is from blue to yellow, their values vary from – 128 to 127. Using this color encoding method slightly simplifies the task for the neural network, since the input image can be used as an L-layer and the neural network will have to come up with only two layers of color, instead of 3, if rgb was used for color encoding.

There are only two possible scenarios of interaction with the telegram bot: the user wrote the command «/start» and the user sent a photo. In response to the first, the bot sends the user the message with information about itself. In response to the sent photo, the bot sends a colorized version converted by neural network. Due to the peculiarities of the neural network, the resolution of the photo sent by the user to the bot necessarily must be a multiple of two, if the user sends a photo of a different resolution, the bot will reply to the user about it and cancel the operation.

If the user sends a color image, it will be converted to black and white and then given to the neural network for colorizing.

Figure 2 shows an example of typing the «/start» command to the bot, as well as a black-and-white photo with a suitable resolution.

Figure 3 shows an example of the bot's response in case of sending it a photo of an inappropriate resolution (1921x1081). The bot sent an error message.

To create and train a neural network, a script separate from the main application was created. After training the neural network, its architecture and weights will be saved in separate files and will subsequently be uploaded into the main application (telegram bot).

The neural network was developed using the tensorflow framework.

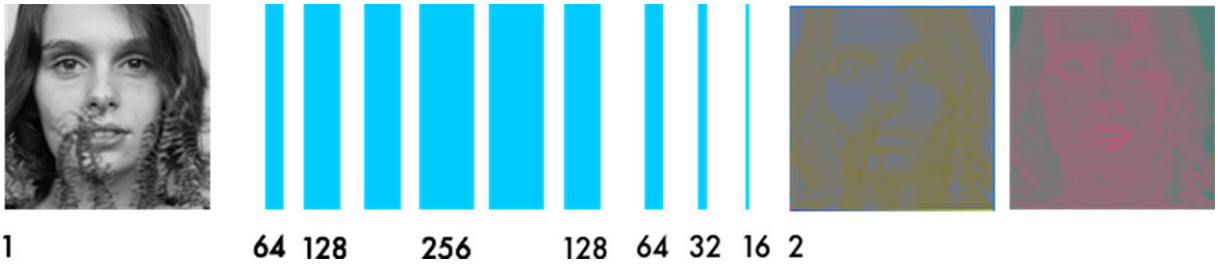

Figure 1 – Architecture of the neural network

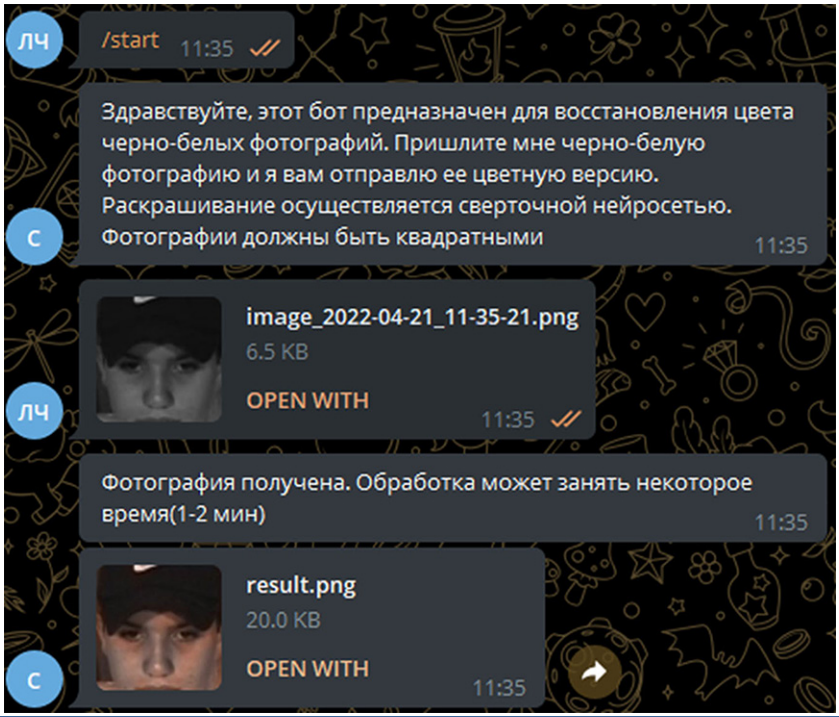

Figure 2 – An example of interaction with the telegram bot

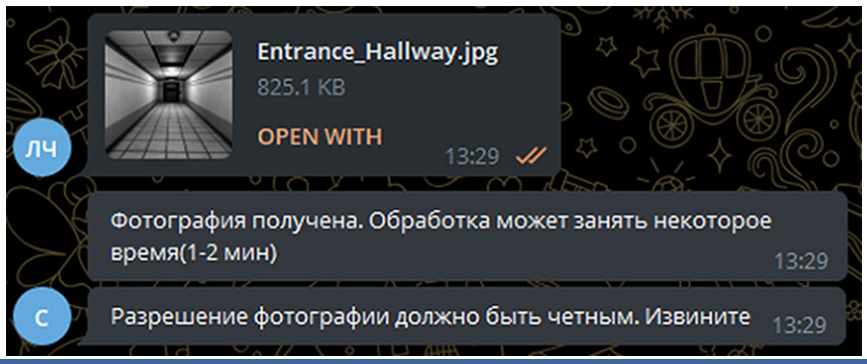

Figure 3 – An example of the bot's response when sending it a photo of inappropriate resolution

```
model = Sequential()
model.add(InputLayer(input_shape=(None, None, 1)))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same', strides=2))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same', strides=2))
model.add(Conv2D(256, (3, 3), activation='relu', padding='same'))
model.add(Conv2D(256, (3, 3), activation='relu', padding='same', strides=2))
model.add(Conv2D(128, (3, 3), activation='relu', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu', padding='same'))
model.add(UpSampling2D((2, 2)))
model.add(Conv2D(32, (3, 3), activation='relu', padding='same'))
model.add(Conv2D(2, (3, 3), activation='tanh', padding='same'))
model.add(UpSampling2D((2, 2)))
model.compile(optimizer='rmsprop', loss='mse')
```

Figure 4 – Neural network implementation in Python using tensorflow framework

Figure 4 shows the source code of the neural network. The input layer is a two-dimensional matrix with an indefinite number of columns and rows so that images of any resolution can be submitted, the matrix is two-dimensional because each pixel of a black-and-white image is described by one number, not three, as in colorized one.

There are no pooling layers in the neural network, since the use of this layer will lead to structural changes in the image, which is unacceptable for colorizing, since the output image must have a completely identical structure to the input one. Instead, we use a stride which is set to two, so that the convolution core will move two pixels horizontally instead of one.

The stride of the convolution core is set by the 'strides' parameter. The final perceptron is also unnecessary, since it is used only in the case of image classification tasks.

The ReLU function (linear rectifier or semi-linear element) was chosen as the activation function of all layers except the output one, since most of the time it shows the best results. In the output layer, we had to use the hyperbolic tangent function, since its ranges from – 1 to 1, unlike ReLU, which has a range of values from 0 to ∞.

This is due to the fact that the color values in the image vary from – 128 to 127. The parameter «padding='same'» is responsible for maintaining the aspect ratio. Before forming the final image, the values of the output layer are multiplied by 128.

In a layer of 256 filters, the final image map is formed and then in the UpSampling2D layers, this map is collapsed back into an image, but now colorized. There are only 2 filters in the last layer. They are just channels 'a' and 'b'.

The python-telegram-bot library was used to develop the telegram bot.

The neural network is loaded into the bot from files that store its structure (model.json) and weights (model.h5). It allows to change the neural network without changing the code of the bot in any way.

Handlers are used in the library to process messages from the user. 3 handlers were created for the bot.

Figure 6 shows the code of creating handlers to respond to the possible scenarios. The first handler responds to the «/start» command, the second handler is triggered when the user sends the bot an image as a picture, the third one as a document; both of these handlers trigger the same function.

Figure 7 shows the source code of the function which is triggered when the user sends the «/start» command to the bot. The function only sends a welcome message to this user.

Figure 8 shows the source code of the function that is triggered by following handlers: photo_as_image_handler and photo_as_document_handler. The bot sends a message to the user about receiving the photo and then first downloads it, then it processes it and sends it as a document to avoid compression from the telegram, since it affects the quality quite a lot.

The photo that the bot downloads is not saved in the hard disk or any other ROM, it is downloaded to a special data type – a byte buffer, which is located in RAM, and prevents access to the photo to anyone or anything other than the bot, and also provides faster access to it.

Scientific results

Testing was done on photos that were included in the training set from the dataset, on photos that were included in the dataset, but not in the training set, and photos that were not included in either the dataset or the training set.

The neural network was trained on 700 photos, where people's faces are captured. The resolution of the photos is 128x128 pixels (Figure 9). Training with more complex data did not lead to satisfactory results.

Based on the results, it can be understood that the neural network copes almost perfectly with photos similar to the dataset.

But with more complex photos where there is something besides faces or any other things which ought to be colorized it barely accomplishes the task.

During the course of programming, the most well-known architectures of artificial neural networks were studied. The neural network architecture that is most suitable for completing the course project task has been selected.

A convolutional neural network has been created and trained. A telegram bot has been created that provided access to the capabilities of the neural network.

Conclusion

The neural network turned out to be capable of colorizing photos which contain faces. Learning from

```
with open('model.json', 'r') as model_layers_file:
    colorise_ai = tf.keras.models.model_from_json(model_layers_file.read())

colorise_ai.load_weights('model.h5')
```

Figure 5 – Uploading the neural network model

```
start_command_handler = CommandHandler('start', start)
photo_as_image_handler = MessageHandler(Filters.photo, handle_photo)
photo_as_document_handler = MessageHandler(Filters.document.category('image') & ~Filters.photo, handle_photo)
```

Figure 6 – Handlers of user messages

```
def start(update:Update, context:CallbackContext):
    context.bot.send_message(chat_id=update.effective_chat.id, text=START_MESSAGE)
```

Figure 7 – Function triggerd by the ‘start_command_handler’

```
def handle_photo(update:Update, context:CallbackContext):
    context.bot.send_message(chat_id=update.effective_chat.id, text='Фотография получена. Обработка может занять некоторое время(1-2 мин)')
    recieved_image_id = get_image_id(update.message)
    img = get_image_from_id(recieved_image_id, context)
    try:
        colorised_photo = colourise_photo(img)
    except ValueError:        You, сейчас • Uncommitted changes
        context.bot.send_message(chat_id=update.effective_chat.id, text='Разрешение фотографии должно быть четным. Извините')
    else:
        context.bot.send_document(
            chat_id=update.effective_chat.id,
            document=colorised_photo,
            filename='result.png',
        )
```

Figure 8 – Function for processing a photo sent by the user

photos with different contents other than faces did not lead to satisfactory results. The neural network also does a poor job with photos that are different from those in the training sample. To correct this, a deeper study of the topic is required, the creation of a neural network with a better configuration and a dataset of photos covering many different subjects. It may be worth exploring the implementation of other

Testing of the neural network

Input photo	Expected result	Output photo
		

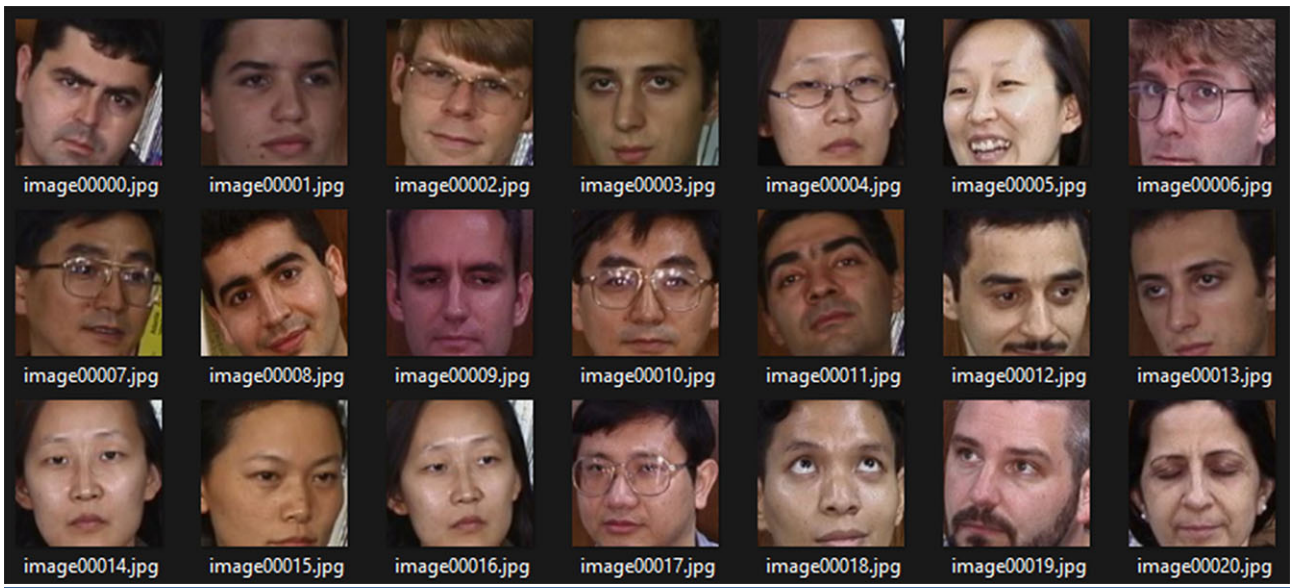

Figure 9 – Some of the images from the training sample

works that solve the same problem.

Advantages of the created application:

- the implementation of the application in the form of a telegram bot provides extremely easy access to the capabilities of the neural network from any device;
- free of charge;
- relatively fast performance.

Disadvantages of the created application:

- the neural network copes well with photos that are similar to those in the training sample;
- there is a limit to the resolution of the photo;
- the neural network is completely unable to cope with anything other than face colorizing.

The scope of machine learning applications is constantly expanding. Ubiquitous informatization leads to the accumulation of huge amounts of data in science, manufacturing, business, transport, and healthcare. The tasks of forecasting, management and decision-making that arise in this case are often reduced to learning from precedents. Previously, when there was no such data, these tasks were either not set at all, or were solved with completely different methods.

REFERENCES

1. Neural networks on Python. Lessons Electronic Resource. URL: https://youtube.com/playlist?list=PLA0M1Bcd0w8yv0XGiF1wjerjZSrYbjh.
2. Kania Alekseevich Kahn. Neural networks. Evolution: SelfPub. 2018. – 288 p.
3. Andrew Trask. Grokaj Deep Learning: 1st edition. – Peter's Publisher. 2019. – 352 p.
4. Tariq Rashid. Creating a Neural Network: 1st Edition. Published by Dialectic Williams. 2019. – 272 p.
5. Paklin, N. B. Oreshkov, V. I. Business analyst: from data to known – yam / N.B. Paklin, V.I. Oreshkov. – Saint Petersburg: Peter, 2017. – 624 p. (3 copies of NB Vol-GU).
6. Kuleichev, A. P. Methods and means of complex data analysis / A.P. Kuleichev. – Moscow: Infra-M, 2016. – 512 p. (20 NB of UlSU).
7. Petrova, E.A. Data Mining: Statistical Methods [Electronic Resource] / E.A. Petrova, A.V. Shevandrin, A.A. Trukhlyaeva. – Volgograd: Consulting, 2018. – 240 p.
8. Kaplan, A. V. Kaplan, V. E. Statistical processing and analysis of economic data /A.V. Kaplan, V.E. Kaplan. – Rostov-on-Don: Phoenix, 2007. – 330 p.
9. Zhogolyev E.A. Object organization of hyperprogramming systems [Text] / E.A. Zhogolyev // Programming. – 2017. – 5. – Pp. 24-32.
10. Kotenko, I.V. Use of multi-agent technologies for complex protection of information resources in computer networks [Electronic resource]: electronic magazine / I.V. Kotenko, O.I. Karsaev. – http:///pitis.tsureru/12.htm., 2015.

Телеграм бот арқылы ақ-қара түсті суреттерді түрлі-түсті форматқа айналдыратын жинақтық нейрондық жүйені қолдану

[1]***МУТОВИНА Наталья Викторовна,*** *т.ғ.к., доцент, mutovina_natalya@mail.ru,*
[1]****АЛИНА Гаухар Жуманжапаровна,*** *магистр, оқытушы, diamond_gaxa@mail.ru,*
[2]***ЮЩЕНКО Олеся Александровна,*** *PhD, т.ғ.к., доцент, РББ бастығы, olessyayuchenko@hotmail.com,*

[1]***ЯНКЕ Никита Анатольевич,*** *студент, diamond_gaxa@mail.ru,*
[1]«Әбілқас Сағынов атындағы Қарағанды техникалық университеті» КеАҚ, Қазақстан, Қарағанды, Н. Назарбаев даңғылы, 56,
[2]Воронеж жоғары технологиялар институты, Ресей, Воронеж, Ленин көшесі, 73а,
**автор-корреспондент.*

Аңдатпа. *Мәселені шешу үшін жасанды нейрондық желілер пайдаланылды, өйткені олар анық емес қосылымдарды анықтау қабілетіне және кіріс және шығыс деректерінің ең жалпы үлгілеріне байланысты болжау үшін ең қолайлы. Құрылған нейрондық желінің архитектурасының диаграммасы әзірленді. Lab түсті кодтау әдісі қарастырылды. Бағдарлама жоғары деңгейлі Python тілінде жүзеге асырылады. Нейрондық желіні құру үшін TensorFlow машиналық оқытуға арналған ашық бағдарламалық қамтамасыз ету кітапханасы пайдаланылды. ReLU функциясы (сызықтық түзеткіш немесе жартылай сызықтық элемент) шығыс қабатынан басқа барлық қабаттар үшін белсендіру функциясы ретінде таңдалды. Жіберілген ақ-қара кескіндерді түрлі-түсті форматқа түрлендірудің негізгі мақсаты болып табылатын жеделхат боты жасалды. Әзірленген бағдарламалық жасақтама алгоритмін қолдана отырып, интернеттегі нейрондық желімен өңдеу арқылы суреттердің төмен сапасы мен шағын ажыратымдылығы мәселесін шешуге болады. Алынған суреттерді кейін сақтауға болады.*

Кілт сөздер: *машиналық оқыту, бот, шығару, Python, нейрондық желілер, түс, хабарлама, чат.*

Использование сверточной нейронной сети для преобразования черно-белых фотографий в цветной формат в телеграм-боте

[1]***МУТОВИНА Наталья Викторовна,*** *к.т.н., доцент, mutovina_natalya@mail.ru,*
[1]****АЛИНА Гаухар Жуманжапаровна,*** *магистр, преподаватель, diamond_gaxa@mail.ru,*
[2]***ЮЩЕНКО Олеся Александровна,*** *PhD, к.т.н., доцент, начальник РИО, olessyayuchenko@hotmail.com,*
[1]***ЯНКЕ Никита Анатольевич,*** *студент, diamond_gaxa@mail.ru,*
[1]НАО «Карагандинский технический университет имени Абылкаса Сагинова», Казахстан, Караганда, пр. Н. Назарбаева, 56,
[2]Воронежский институт высоких технологий, Россия, Воронеж, ул. Ленина, 73а,
**автор-корреспондент.*

Аннотация. *Для решения задачи использованы искусственные нейронные сети, так как они лучше всего подходят для предсказаний, благодаря их способности обнаруживать неочевидные связи и самые общие закономерности входных и выходных данных. Разработана схема архитектуры созданной нейронной сети. Рассмотрен метод цветового кодирования Lab. Программа реализована на языке высокого уровня Python. Для создания нейронной сети использована открытая программная библиотека для машинного обучения TensorFlow. В качестве функции активации всех слоев, кроме выходного, была выбрана функция ReLU (линейный выпрямитель или полулинейный элемент). Разработан телеграм-бот, основная цель которого – преобразование отправленных черно-белых изображений в цветной формат. С помощью разработанного программного алгоритма можно также решить проблему низкого качества и небольшого разрешения изображений, путем их обработки нейронной сетью онлайн. После полученные изображения можно сохранить.*

Ключевые слова: *машинное обучение, бот, выходные данные, Python, нейронные сети, цвет, сообщение, чат.*

REFERENCES

1. Neural networks on Python. Lessons Electronic Resource. URL: https://youtube.com/playlist?list=PLA0M1Bcd0w8yv0XGiF1wjerjZSrYbjh.
2. Kania Alekseevich Kahn. Neural networks. Evolution: SelfPub. 2018. – 288 p.
3. Andrew Trask. Grokaj Deep Learning: 1st edition. – Peter's Publisher. 2019. – 352 p.
4. Tariq Rashid. Creating a Neural Network: 1st Edition. Published by Dialectic Williams. 2019. – 272 p.
5. Paklin, N. B. Oreshkov, V. I. Business analyst: from data to known – yam / N.B. Paklin, V.I. Oreshkov. – Saint Petersburg: Peter, 2017. – 624 p. (3 copies of NB Vol-GU).
6. Kuleichev, A. P. Methods and means of complex data analysis / A.P. Kuleichev. – Moscow: Infra-M, 2016. – 512 p. (20 NB of UlSU).
7. Petrova, E.A. Data Mining: Statistical Methods [Electronic Resource] / E.A. Petrova, A.V. Shevandrin, A.A. Trukhlyaeva. – Volgograd: Consulting, 2018. – 240 p.
8. Kaplan, A. V. Kaplan, V. E. Statistical processing and analysis of economic data /A.V. Kaplan, V.E. Kaplan. – Rostov-on-Don: Phoenix, 2007. – 330 p.
9. Zhogolyev E.A. Object organization of hyperprogramming systems [Text] / E.A. Zhogolyev // Programming. – 2017. – 5. – Pp. 24-32.
10. Kotenko, I.V. Use of multi-agent technologies for complex protection of information resources in computer networks [Electronic resource]: electronic magazine / I.V. Kotenko, O.I. Karsaev. – http:///pitis.tsureru/12.htm., 2015.