Bitmap Image Recognition with Neural Networks

Dmytro Uchkin, Tetyana Korotyeyeva, Tetiana Shestakevych

Lviv Polytechnic National University

dmytro.uchkin.pz.2016@lpnu.ua, tetyana.o.korotyeyeva@lpnu.ua

Received: May 11, 2020, accepted June 01, 2020

Abstract. Logistics, finance, science, and trade are just some of the areas that require computer vision technology, which includes number recognition. The need to recognize numbers in images or photographs is found in tasks such as recognizing car numbers, reading values from paper bills, recognizing object identification numbers, and reading credit card numbers. The development of an online application for recognition numbers in bitmap images using machine training technologies, namely an artificial neural network based on the class of neural networks perceptron, is an actual task.

Keywords: neural network, digitized image

INTRODUCTION

The processes of automatic reading and identification of symbols or images have become the norm in many areas of human activity. The spread of these processes is facilitated by the increase of visual fixation devices on and off the roads of the city, the improvement of visualization in medicine, as well as, among other things, the constant need to read codes in trade and text recognition in documents.

Among the open access sources indexed in the Scopus bibliographic database, by keywords *image recognition* are about three thousand articles from various fields, the top 10 of which are Computer Science, Engineering Mathematics, Physics and Astronomy, Materials Science, Biochemistry, Genetics and Molecular Biology, Medicine, Chemistry, Environmental Science, Social Sciences. In the found articles the phrase *text recognition* occurs in works from the following fields: Computer Science, Engineering, Materials Science, Physics and Astronomy, Agricultural and Biological Sciences, Mathematics, Neuroscience, Chemical Engineering, Environmental Science, Multidisciplinary, Business, Management and Accounting, Energy, Social Sciences. To confirm the breadth of the fields in which text recognition processes are used, we note that in physics and astronomy, the test recognition subsystem in the Airborne Display and Control System uses neural networks [1] to image recognition in text processing, highlighting, classification, and correction. Iterative methods were used to recognize handwritten Chinese characters [2]; embedded networks was used to recognize images of Korean characters [3]; in [4], the authors suggested the line-segment feature analysis algorithm to handwritten text recognition; to recognize Amharic (an indigenous Ethiopic script) a group of authors implemented an end-to-end trainable neural network [5]. A computational model for printed text images explained in [6], and a variant of Recurrent Neural Network was used to formulate the character image recognition model.

Using different kinds of neural networks, the monitoring of coral reefs was improved [7], as well as the finding of the center for the pivot irrigation systems [8], and to recognize the type of down (in Agriculture) [9].

Digital image processing is performed with different approaches [10-14]. In [15] authors stated, that Chinese chess playing can help prevent Alzheimer disease, and the robotic technology for such an activity needs an image recognition subsystem. Special features of helping people with visual impairments using popular information technologies were investigated in [16].

IMAGE RECOGNITION SOFTWARE: ADVANTAGES AND DISADVANTAGES

Automatic online handwriting recognition has been a topical research issue for four decades now and remains so up to this day. Many mobile applications allow drawing on the screen, manually or with a stylus, but only a few allow you to recognize and digitize the data entered. Examples include MetaMoJi Note, Notes Plus, MyScript Nebo, WritePad for iPad, Mazec, GoodNotes 5, which starts at $ 5, and the free Google Handwriting Input and Pen to Print apps. These applications recognize English text, so they cannot be used to recognize Ukrainian written text.

As software for numbers recognition, it is worth mentioning Optical Number Recognition tool (https://www.softpedia.com/get/Office-tools/Other-Office-Tools/Optical-Number-Recognition.shtml), and Recog for Windows.

Among the advantages of the Optical Number Recognition should be mentioned the followed.

The application offers the ability to manually draw the target number directly in the main window. You can start drawing by left-clicking and erase unnecessary parts by right-clicking (Fig. 1).

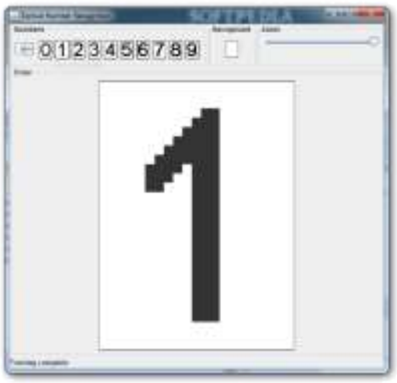
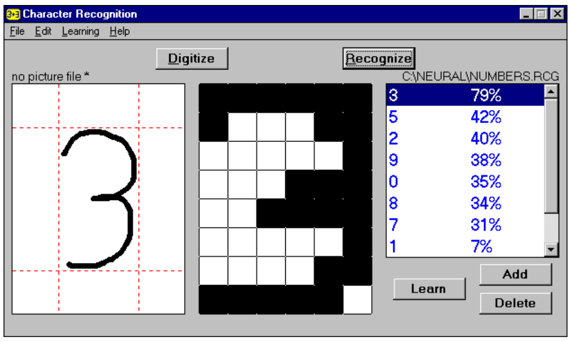


Fig. 1. *Optical Number Recognition* program interface

- The application allows zooming in the image, as well as select the numbers you want to recognize.
- The application is resource-friendly, so its overall performance is not significantly reduced.

The advantages of the Recog for Windows application are as follows:

- The application can be trained to recognize a new character.
- The accuracy of the results increases when it was rained with a new element.
- You can see the difference between controlled and uncontrolled training (Fig. 2).



```
3   79%
5   42%
2   40%
9   38%
0   35%
8   34%
7   31%
1   7%
```

Fig. 2. *Character Recognition* application interface

The disadvantages of both applications are the lack of a web version, which makes it impossible to recognize over the Internet, and outdated interface.

FEATURES OF NEURAL NETWORKS APPLICATION

The idea of designing intelligent computing devices by the image and likeness of biological systems led to the creation of the theory of neural networks [17], which became one of the most powerful and useful approaches to the development of artificial intelligence. If we talk about today's achievements, for example, neural networks are the main activity of Deepmind (now owned by Google), which has achieved such fantastic success as creating a neural network capable of training to play video games and another that was able to win in an incredibly difficult *Go* game over a world-class grandmaster.

The artificial neural network training algorithm [17], or *neural network* (NN), is a training algorithm remotely inspired by biological neural networks. Computations are structured in terms of interconnected groups of artificial neurons that process information using a connectivist computing approach [18].

Modern neural networks are nonlinear statistical data modeling tools. They are most often used to model complex relationships between inputs and outputs, to find patterns in the data, or to identify the statistical structure in the unknown common probability distribution of the observed quantities.

Observing how a neuron works, we find that it receives an electrical signal that enters and stimulates another electrical signal. This is similar to the work of predictor models classifier, which receive some input and perform certain calculations to return the result. According to the results of observations, neurons do not react instantly but suppress the input signal until it reaches the value that will start the generation of the output signal. This can be imagined as the presence of a certain threshold value that must be exceeded before the output pulse is generated.

A function, based on a threshold value, that receives a signal at the input and returns an output signal, is called an activation function. From a mathematical point of view, many such functions could provide a similar effect. An example is the step function, sigmoidal function, etc. [19].

An algorithm based on an artificial neural network will be used to recognize numbers in bitmap images. This software solution will be advisable to present in the form of a library that will add flexibility to the software, and allow it to be integrated into other application systems, i.e. software for postal services to recognize numbers on envelopes and parcels, automatic control systems, and exam works of students, in collecting works of art, etc.

FUNCTIONS OF BITMAP IMAGE RECOGNITION SYSTEM

The designed system should provide training of the neural network on an array of bitmap images of numbers and their subsequent recognition.

The operating environment of the developed software is the operating system *Windows*, *macOS*, or *Linux*; one must also install the *Jupyter* environment and the *Python* interpreter.

To train the network by loading the input array, the user sequence is as follows:

1. The user opens the web application.
2. The user loads an array of data from a file.
3. The user enters the number and options of network training iterations.
4. The user launches a training.
5. The user checks the effectiveness of the training with arbitrary test data.

6. The user saves the training configuration if the NN training result is satisfactory.

The number of such training is unlimited, the training process of the neural network can be improved. The user can download bitmap images of any format.

The software also provides the ability to *write* a number on a canvas interface element, mimicking handwritten text for further recognition. The sequence of actions for this:

1. The user opens the web application.

2. The user goes to the *writing* area and puts down a number.

3. The user initiates the recognition of the number.

4. The user receives a text recognition result.

The user can select the size of the input area.

The user also has the function of copying the recognition results to the clipboard, in particular, using a keyboard shortcut.

The user interface will be presented as a web page (web application), so all the basic requirements for the external interface of any web page are valid for this interface, including:

- Adaptability, i.e. compatibility with the needs and capabilities of the user, ease of transition between functions, mutually complementary forms of results presentation;
- Sufficiency of the interface, when system user requests must be unambiguous and clear for users of all levels, as well as for application tasks of all classes, the system's response to any type of request must also be unambiguous and clear;
- Flexibility of the interface, i.e. the ability to adapt to a specific task.

The technical requirements for the interface are:

- Download speed, i.e. the application interface must be loaded in no longer than 1s;
- Adaptability and sensitivity, i.e. the interface of the application should look the same on different extensions and screen sizes, regardless of the device;
- Accessibility, i.e. the interface must be accessible for use by people with visual and hearing impairments [20].

The application should not depend on hardware interfaces. The web interface is implemented by the HTTP application layer network protocol and the AJAX standard.

The designed application does not have security requirements, because it does not require user`s personal data.

NEURAL NETWORK IMPLEMENTATION TOOLS

The application is developed for the *Jupyter* environment, so it is advisable to use the *Python* language for development. *Python* is a programming language that supports module packages and individual modules. The language supports several programming paradigms: object-oriented, procedural, functional.

The developed web application consists of a server and client components, implementing the *client-server* architecture. The front-end party implements the user interface, generates requests (HTTP requests) to the server, and processes responses from it. The back-end part of the system processes the request from the front-end part, performs calculations, then generates a web page and sends it to the client over the network using the HTTP protocol.

Microsoft's free *Visual Studio Code* product *IDE* will be used to develop the code, to test, and to improve the software. *Google Chrome* will also be used as a testing and launching environment for the web page.

THE ARCHITECTURE OF THE DESIGNED NEURAL NETWORK

The architecture of a neural network is determined primarily by the task of this neural network. Taking into account that the task is to create a neural network to recognize numbers in bitmap images, so this problem can be formulated as a problem of classifying objects in the image, namely numbers from zero to nine. Based on this, we can proceed to the selection of the class (architecture) of the neural network.

The perceptron is constructed of three different types of elements that interact as follows: impulses coming from the receptors (elements of the first type) pass to the associative elements (elements of the second type), and then to the responding neurons (the third type). In this way, perceptrons allow creating some *associations* between the input stimuli and the required response at the output. From a biological point of view, this corresponds to the transformation, for example, of visual information into a physiological response of motor neurons. According to modern terminology, perceptrons can be classified as artificial neural networks [21]:

- with one hidden layer;
- with threshold transfer function;
- with feed forward signal transmission.

Based on the above facts, the neural network acquires the following architectural features:

1. **Number of layers** is 3: input (touch), hidden (associative), output (responsive);
2. **Number of nodes** in the input layer *n*m*, where *n* is the number of image pixels vertically, and *m* is the number of pixels horizontally; 10 nodes on the source layer, which is equal to the number of possible classes of objects, namely the numbers from 0 to 9.
3. **Activation function** is sigmoid, as the most common and effective function in terms of training speed.

DESIGNING OF BITMAP IMAGE RECOGNITION SYSTEM USING NEURAL NETWORKS

A class diagram is a static representation of the model structure, which demonstrates stable (declarative) elements, such as classes, data types, their content, and relationships. The class diagram may include notation for packages and may contain notation for nested packages. It should be noted that the class diagram may also include the designation of some elements of behavior, but their dynamics are manifested to a greater extent in other types of UML diagrams. This type of diagram represents the ontology of the domain and is similar in content to the information model according to the method of S. Schleier and S. Mellor: it determines the content of object classes as basic abstractions and their relationships. Moreover, the notation for class description provides the separation of function description from data description, the use of encapsulation paradigms, and data inheritance.

The neural network will contain only one NeuralNetwork class, which will have three methods:

- **initialization**: setting the number of input, hidden and output nodes;
- **training**: calibration of weights in the dynamics of processing of the training samples presented for training;
- **survey**: receiving signal readings from output nodes after entering the values of input signals.

Because the neural network does not use any data for its operation, it does not need a database. A file will be used to store the results (nodes) of the network training, and the intermediate calculations will be stored in RAM only during the operation of the neural network process.

To model the processes of system components interaction, ordered in time, it is advisable to use a UML sequence diagram.

Sequence diagrams typically contain objects that interact within the boundaries of the script, the notifications they exchange, and the results returned, with the messages concerned.

The sequence diagram shows only those objects that are directly involved in the interaction.

The graphical user interface will be developed using *Ant Design* components that meet the requirements for developing graphical interfaces, defined by Google. Prototyping was chosen as the most flexible and simple method because it does not require knowledge of graphic editors, only a program that allows creating interfaces from ready-made blocks.

NEURAL NETWORK IMPLEMENTATION AND TESTING

We implement the neural network by three methods that will reflect its functionality (described earlier): initialization (constructor), training, and survey. The initial step is to implement the initialization (constructor) of the neural network class, which includes the need to specify the number of nodes of the input, hidden, and output layers. This data will determine the configuration and size of the neural network. Instead of rigidly prescribing these values in the program code, it will be necessary to set these values as parameters in the process of creating a network instance. As a result, it will be easy to create new neural networks of different scales by adapting them to images of different sizes.

In addition, it is necessary to be able to set the training factor at the initialization stage. This setting can also be set when creating a new neural network.

The next step is to create a network that consists of nodes and connections. Weights are important in the neural network. They are used to calculate the spread of signals in the forward direction, as well as the reverse spread of errors, it is the weights are specified in an attempt to improve network performance.

These weights are an integral part of the neural network and serve as its characteristics throughout its life. This is because they must be a part of the initialization process and be available for other methods, such as network training, and functions survey. It should be noted that the initial values of the weights should be small and selected at random.

The next step in the implementation of the neural network is the implementation of the method of interviewing the neural network when the input data for recognition is provided and the answer is obtained.

The neural network survey method receives the input signals of the neural network as parameters and returns its output signals. To do this, it is necessary to transmit signal data from the nodes of the input layer through the hidden layer to the nodes of the output layer. Thus, as the signals spread, they need to be aligned using the weights of the relationships between the respective nodes and use the sigmoid function to smooth the output pulses of the nodes.

Among the three main methods of the neural network, the most important is the method of training (calibration) of the neural network, which is the main process in creating neural networks and is responsible for data classifying. The neural network training process can be divided into two parts.

- The first part: the calculation of output signals for a given training example. This functionality is no different from that underlying the neural network survey method.
- The second part: comparing the calculated output signals with the expected response and making changes to the weights of the relationships between nodes based on the identified differences.

The next step in neural network training is to refine the weights based on the difference between the calculated and target values.

First of all, it is necessary to calculate the error, which is the difference between the desired target output value provided by the training example and the actual output value.

Next, you need to calculate the inverse error spread for the hidden layer nodes. To do this, you need to distribute the errors between the nodes in proportion to the weights of the bonds, and then recombine them at each node of the hidden layer.

The last step is to refine and update the weights, which is based on the gradient descent method and implemented in matrix form.

NEURAL NETWORK TRAINING

A neural network class implementation is a process of describing a model that will allow numbers recognition in images, but a neural network model alone cannot reliably recognize numbers without a training step. The process of neural network training is a cyclic process, which is based on improving the model by obtaining a large array of data at the input, calibrating the coefficient, and testing the network on test data.

There is a collection of images of handwritten numbers used by artificial intelligence researchers as a popular set for testing ideas and algorithms. This test suite is a database of handwritten numbers called MNIST, provided by authoritative neural network researcher Jan Lekun for free public access at http://yann.lecun.com/exdb/mnist/. There you can also find information on the success of past and present attempts to correctly recognize these handwritten characters.

The format of the MNIST database is not one that is easy to work with, but other experts have created the appropriate files in a simpler format. These are so-called CSV files, in which individual values are plain text, separated by commas. Their contents can be easily viewed in any text editor, and most spreadsheets or data analysis programs can work with CSV files. This is a fairly universal format.

The training set contains 60,000 marked copies that are used to train the neural network. Here *marked* means that for each instance the corresponding correct number is specified.

A smaller test set of 10,000 copies is used to verify that ideas or algorithms are working correctly. It also contains correct markers to see if the neural network is able to give the correct result.

The use of independent training and test data sets ensures that the neural network has not previously encountered test data. Otherwise, the neural network would simply memorize training data and not provide reliable statistics on the success of recognition. The idea of separating training and test data is common among machine training professionals.

The basic resolution of such a model after training on 60,000 records is 94.73%, ie an error rate of 5.27%, which corresponds to the reference values. The description of obtaining the recognition success rate and its improvement is described in the next section.

STUDY OF THE NEURAL NETWORK

To test the efficiency of a neural network, one must first determine the metrics and properties that affect this metric for further measurement and calculation. Such a metric for a neural network for object classification is *recognition efficiency*, which reflects the number of correctly recognized objects to the total number of such objects. When the MNIST test data set contains 10,000 records, the number of correctly recognized digits is divided by the total number of records and will be the recognition efficiency factor.

When testing the neural network with the above code on the test data MNIST, containing 10,000 records, the result is 0.9473. Thus, the efficiency of neural network recognition is 94.73%.

It is also necessary to identify factors that affect the efficiency of the neural network. For the network of this architecture, the main factors are the number of training epochs (Fig. 3), the number of hidden layer nodes, the training factor (Fig. 4).

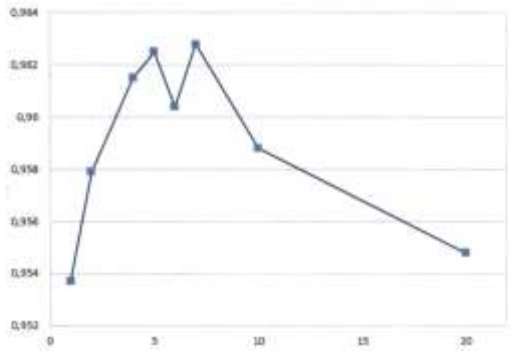


Fig. 3. Graph of the dependence of recognition efficiency on the number of epochs

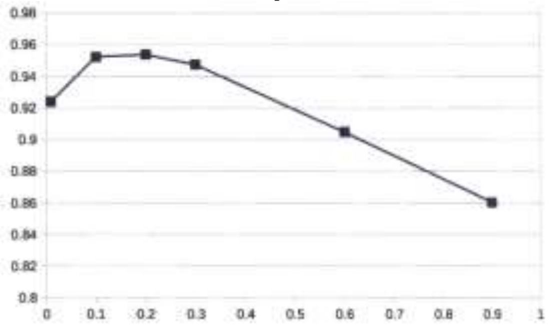


Fig. 4. Graph of the dependence of the recognition efficiency on the training factor

Based on the data presented in the graphs above, we can conclude that with the number of epochs equal to five and a training factor of 0.1, the neural network reaches the peak of recognition efficiency, which is 96.25%.

The testing phase makes it possible to detect errors in the system. Thus, it can be argued that the developed system meets the established requirements. Testing of the graphical user interface is carried out using the developed test cases. All results must meet expectations.

CONCLUSIONS

The current level of development of digital technologies makes it possible to create and use tools for working with digital images. Such processes include digitalization of images and text, both printed and written, recognition of such text for different needs. There are software products that in one way or another allow digitizing bitmap images of numbers, however, such software, along with the advantages, also has some of disadvantages. To avoid such shortcomings, appropriate software has been designed for which functions are specified. The developed software uses machine training methods (based on an artificial neural network) to recognize handwritten numbers, provides the ability to integrate with other software. The Python language in the Jupyter environment was used to develop the software. Interaction with the user takes place through a web application. The neural network of the developed software was trained on the normalized base of handwritten numbers MNIST, the exceeding of the reference indicator of the efficiency of recognition among similar systems on recognition of numbers was fixed.

One of the areas of further development and improvement of the software is the possibility of its use for image recognition for the needs of connoisseurs of antiques. For example, the marks on the dishes (Fig. 5) can determine the manufacturer, the approximate time of creation, which will allow you to estimate the approximate cost of the product.

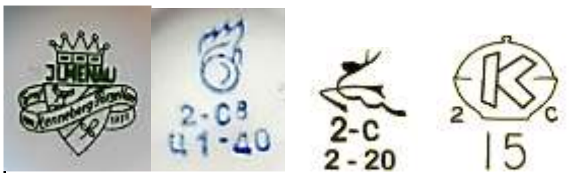


Fig. 5. Brands of dishes from the times of the Soviet Union

In this way, the improved software will be relevant for antiquarians, researchers interested in different historical epochs, collectors, and more.

REFERENCES

1. **He J., Li X., Xiao J. 2020.** CNN-based image recognition and processing technology development of airborne display and control system, Journal of Physics: Conference Series. 1684 (1), art. no. 012108.
2. **Yu W., Li Y., Peng H., Zhang L. 2020.** Image iterative method for handwritten Chinese character recognition, Journal of Physics: Conference Series, 1684 (1), art. no. 012101.
3. **Ilyuhin S. A., Sheshkus A. V., Arlazarov V. L. 2020.** Recognition of images of Korean characters using embedded networks, Proceedings of SPIE, The International Society for Optical Engineering, 11433, art. no. 1143311.
4. **Kim C.-M., Hong E.J., Chung K., Park R.C. 2020.** Line-segment feature analysis algorithm using input dimensionality reduction for handwritten text recognition, Applied Sciences (Switzerland), 10 (19), art. no. 6904, pp. 1-17.
5. **Belay B., Habtegebrial T., Meshesha M., Liwicki M., Belay G., Stricker D. 2020.** Amharic ocr: An end-to-end training, Applied Sciences (Switzerland), 10 (3), art. no. 1117.
6. **Oni O. J., Asahiah F. O. 2020.** Computational modelling of an optical character recognition system for Yorùbá printed text images, Scientific African, 9, art. no. e00415.
7. **Raphael A., Dubinsky Z., Iluz D., Netanyahu N. S. 2020.** Neural network recognition of marine benthos and corals, Diversity, 12 (1), art. no. 29.
8. **Zhang C., Yue P., Di L., Wu Z. 2018.** Automatic identification of center pivot irrigation systems from landsat images using convolutional neural networks, Agriculture (Switzerland), 8 (10), art. no. 147.
9. **Yang W., Liu, Q., Wang S., Cui Z., Chen X., Chen L., Zhang N. 2018.** Down image recognition based on deep convolutional neural network, Information Processing in Agriculture, 5 (2), pp. 246-252.
10. **Rybchak Z., Basystiuk O. 2017.** Analysis of computer vision and image analysis technics. ECONTECHMOD, vol. 6. No. 2. Pp. 79–84.
11. **Shumeiko A., Smorodskyi V. 2017.** Discrete trigonometric transform and its usage in digital image processing. ECONTECHMOD, vol. 06, No. 4, pp. 21-26.
12. **Kaminsky R., Borovets Ya. 2017.** Model of operator activity in computer systems image processing. ECONTECHMOD, vol. 6. No. 2, pp. 9–14.
13. **Veres O., Kis Ya., Kugivchak V., Rishniak I. 2018.** Development of a Reverse-search System of Similar or Identical Images. ECONTECHMOD, vol. 07, No. 2, pp. 23-30.
14. **Boyko N., Sokil N. 2017.** Building computer vision systems using machine training algorithms, ECONTECHMOD. vol. 6, No. 2, pp. 15–20.
15. **Chen P.-J., Yang S.-Y., Wang C.-S., Muslikhin M., Wang M.-S. 2020.** Development of a chinese chess robotic system for the elderly using convolutional neural networks, Sustainability (Switzerland), 12 (10), art. no. 3980.
16. **Paulino D., Reis A., Paredes H., Fernandes H., Barroso, J. 2019.** Usage of artificial vision cloud services as building blocks for blind people assistive systems, International Journal of Recent Technology and Engineering, 8 (2 Special Issue 10), pp. 453-458.
17. **Matvieyeva Ye. 2016.** Algoriths: development and application, Classics of Computers Science, StPb, 800 p.
18. **Coreman T., et al. 2019.** Algorithms: design and analysis, Dialectica, StPb, 1328 p.
19. **Rafaello C. 2013.** Graphics on JavaScript, StPb, 272 p.
20. **Shestakevych T., Pasichnyk V., Kunanets N., Medykovskyy M., Antonyuk N. 2018.** The content web-accessibility of information and technology support in a complex system of educational and social inclusion, 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies, CSIT 2018 - Proceedings, 1, art. no. 8526691, pp. 27-31.
21. **Simpson K. 2015.** You don’t know JS: UP & Going, O’Reilly Media, Inc, 224 p.