International Journal of Computer Science and Engineering (IJCSE)
ISSN(P): 2278–9960; ISSN(E): 2278–9979
Vol. 10, Issue 1, Jan–Jun 2021; 1–10

International Academy of Science, Engineering and Technology
Connecting Researchers; Nurturing Innovations

CREATION OF ARAB GRAPHIC WRITINGS RECOGNITION PROGRAM

Turaqulov Sh. X[1] & Iskandarova S.N[2]

[1]Head of the Technical Support Department of the Information Technology Center of Samarkand State University, Samarkand, Uzbekistan

[2]Research and Innovation Center for Information and Communication Technologies at the Tashkent University of Information Technologies Named after Muhammad al Khorezmi, Tashkent, Uzbekistan

ABSTRACT

The most effective results in the processing and recognition of images are achieved through the development of information technology, the introduction of new technologies. The TensorFlow library capabilities in the python programming environment are huge in getting effective results. It describes how to use open data sets to recognize Arabic graphics and to form datasets for additional letters for old Uzbek and Farsi text letters. Examples, advantages and analysis of the results obtained by the TensorFlow platform to perform calculations with python are given. The efficacy and mode of use of the convolutional neural network are described. Results were obtained from the use of open data sets via www.kaggle.com. The most effective methods of recognizing the given Arabic text are used, and only the results obtained are described, without the algorithms given in the references. CNN created a model for the data set letters, based on which the results were 90% recognizable.

KEYWORDS: *Segmentation, CNN (Convolutional Neural Network), Tenserflow, Python, Neural Network, Recognition, Model*

Article History

Received: *02 Dec 2020* **|** ***Revised:*** *04 Dec 2020* **|** ***Accepted:*** *22 Dec 2020*

INTRODUCTION

There are many ready-made libraries for working with neural networks for the Python programming language. The use of these libraries is designed to significantly simplify the task of developing neural networks. TensorFlow [2] is a very popular library developed by Google. The documentation for this library is well developed. There are many articles on the internet and examples of neural networks for this library. Here, graphs are used to represent a neural network, and multidimensional rows, tensors, are used to store data. Like other libraries, it has integration with the NumPy library. Both the CPU and the GPU have the ability to perform calculations. The Keras library is, more precisely, the basis for the creation of neural networks. In addition to Keras TensorFlow and Theano, Keras can use one of these two libraries to perform calculations. Of all the libraries, this is the most convenient and easy to understand. Here new layers are added using only one function. Good documentation and lots of ready-made examples - all of which are a big advantage for Keras. The development of these capabilities has the great advantage of recognizing manuscript sources with complex forms. Arabic graphics have a complex shape, and the fact that its 28 letters appear in different forms in the middle and at the end of a word, and that they form a different shape due to the combination of letters, causes some problems in

recognition. In solving these problems, we analyze the results using the following segmentation algorithms and neural networks in recognition.

Methodology of Segmentation of the Arabic Text

We will be able to make effective use of neural network capabilities and analyze the results in a python programming environment. We use horizontal segmentation to recognize each line and vertical segmentation to separate words and letters.

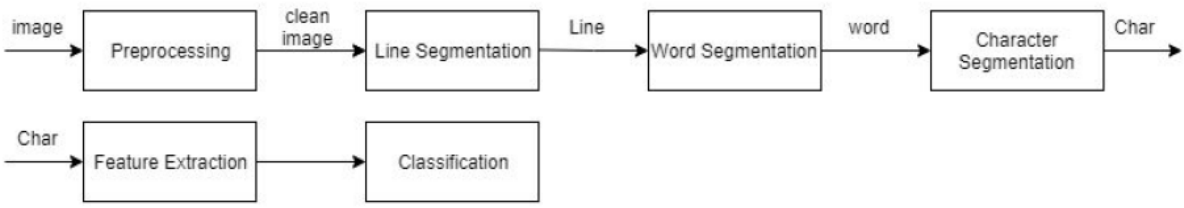

Figure 1: Segments in Recognition.

How to add text to letters in Arabic, these three segmentations are done using word segmentation. Line segmentation approaches: Projection-based approach; when assembled horizontally, this is called the horizontal projection along the vertical axis for each x value, and this is called the vertical pro18]. Smearing approach; text strings calculated within a predetermined boundary between the white space along the horizontal line are bounded by copies [6, 10, 11]. Grouping method; it is built by grouping adjacent Arabic text such as "ﺛ, ﺷـ", entries such as "ﺗ, ﻳـ". The problem is that projection-based words and characters use a simple constraint tool during character segmentation. It is based on certain cognitive criteria such as similarity, continuity, and closeness [7, 10]. Word and Character Segmentation Approaches: There are four main approaches to linking characters in an Arabic word [1].

- Assume that the input is already divided into characters (there is no need for character segmentation).
- Divide the introductory words into smaller primitive parts than the character, and then group each elementary group into characters as they are identified.
- Divide words into characters. This is the most difficult approach in the language of nature.
- Recognize keywords without segmentation in general.

Histogram-Based Algorithms Several algorithms use histogram-based methods in different languages for OCR. None of these algorithms solve all the problems associated with OCR in Arabic. In Arabic, only the division into linear and connected parts is done, and the most important thing in Arabic is to separate these characters, and this is ignored [20].

Proposed Algorithm This document belongs to a segmentation order that accepts an Arabic text image and outputs separated characters. It consists of three stages. The first is responsible for identifying and separating lines in the text. The segmented lines then skip the second step, which is to retrieve words from the text lines. Finally, the third stage takes these words and creates a characteristic appearance of each word. The proposed algorithm considers the nature of the character between the characters and partially overlaps .The previous step is used to detect and correct the curvature of the scanned text image. The algorithm in [15] is used to perform the angle of rotation correction.

Line splitting: Line segmentation is performed using the Image Axis Profile method, which calculates the horizontal axis profile for a binary text image [10].

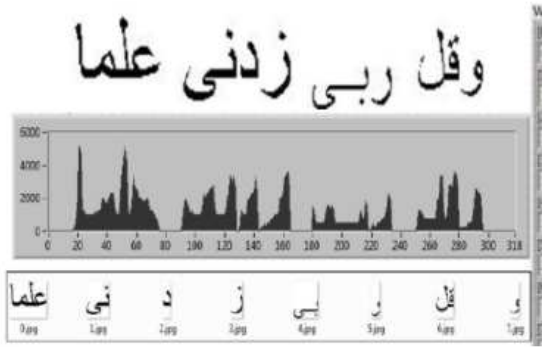

Figure 2: Separate Words from a Line.

Figure 2 shows an image of an Arabic text line and a corresponding vertical profile. The text line is divided into parts that are connected from left to right. These connected parts are grouped into the appropriate word. Each word is an introductory image of the character segmentation phase.

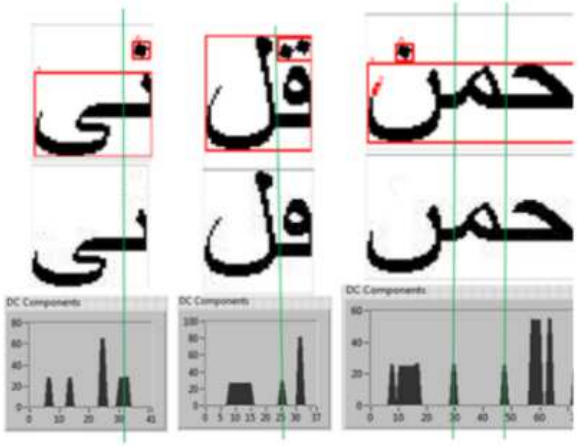

Figure 3: Characterization Procedure.

Image highlighting is used to identify a pre-configured data set using a model taught in CNN. In recognizing these manuscripts, the data set used an open data set available at https://www.kaggle.com for the Urdu letter, and the results were obtained.

Procedure of Using CNN

A distinctive feature of the CNN neural network is that first-level neurons are arranged in a special structure: in the first layers, neurons are divided into images of a certain size (sometimes called cards), and different cards within the same layer represent neurons that respond to different image characteristics. Corresponds to different species [3]. There are two types of calculating the activation of the next layer in CNNs. The first type: activation of neurons at the next level is considered as a linear combination of activation of neurons at the previous level, and the weight of these linear activities depends only on the relative position of neurons, types of neurons, but they do not depend on the state of a particular neuron on the map. The second type: the activation of neurons at the next level simply replicates the activity of neurons at the previous level, but the activation of neighbouring neurons because they are replaced by their maximum or average values is called the

process of reduction. This structure makes it very convenient for CNN to work with images because it allows, for example, if two images differ by a small shift, the network achieves a very similar result at the same time as in normal, non-convulsive neural networks. This is not the right result. In addition, the number of parameters in convulsive artificial neural networks is smaller than the number of neurons. Typical neural networks for the same number of neurons can have hundreds of billions of parameters, and it is impossible to include training kits to train such a large number of parameters, there are aggregate neural networks with so many neurons can be taught in samples and this ensures a high result in the recognition of manuscript texts [2].

A review of ready-made solutions for the creation of neural networks, the first CNN (Convolutional Neural Network) neural network for character identification, was introduced in 1998 by French researcher Yann LeCun [1]. It's called LeNet. The structure of this network is shown in the following figure.

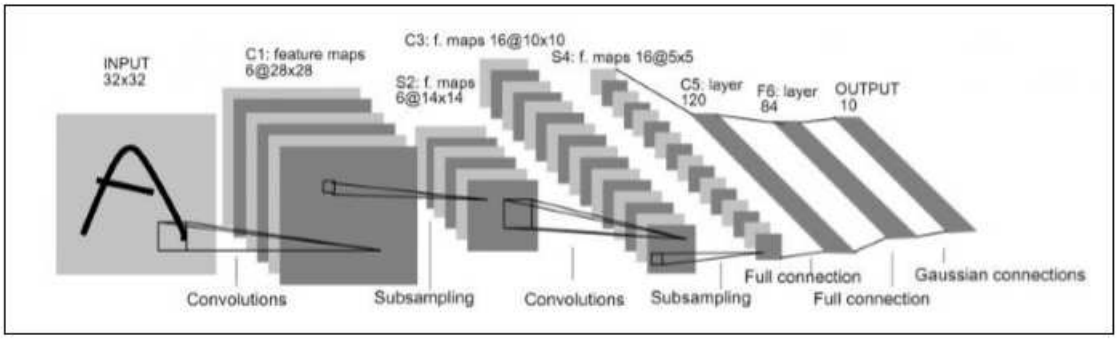

Figure 4: The Structure of the LeNet Neural Network.

The general calculation procedure of CNN was chosen as follows:

CNN=40:3x3 pool 2x2, CNN=60:3x3, pool=2x2, lstm=200, drout=0.5. ctc loss

Recognition software developed in python. Here is the identification procedure based on the code. Here 40 filters and a 3x3 pointer are the core function and we use one pool = 2x2. We use 60 filters 3x3 core function, pool = 2x2 i.e. 2 CNN. To get small input settings.

The CNN structure serves to obtain glyphs of given words from a series of segmented images. The series image is large in size and can be swiped 40 times using the 3x3 core function and the pool 2x2. In the next CNN section, a swipe operation is performed with 60 3x3 core functions and pool = 2x2. The CNN neural network output class is the number of rows. In an interconnected neural network, a multilayer perceptron neural network was used and classified using inner layers with 16 neurons and a relay activation function.

On CNN, input neurons are numerical representations of a series of images:

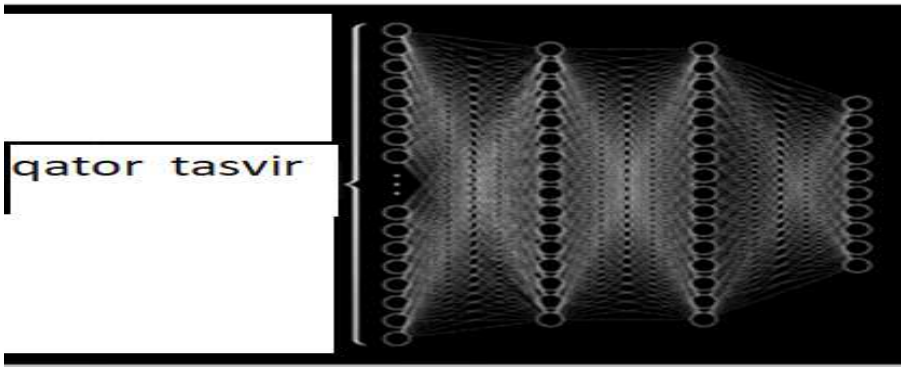

Figure 5: Image Processing in Neurons.

If the image size is large, the number of input neurons will increase accordingly. For example, in an image measuring 28x28 pixels, the number of input neurons is 784, which is 13,002 internal neurons in a multilayer perceptron neuron. At CNN, we can use a kernel matrix to extract important features of an image, and we can have an active matrix using a pool swipe operation. Using this active matrix, we can store the base parameters of the classes of this shape. This led to a contraction of the input neurons given to us. An example of a symbol is:

Figure 6. Different Aspects of Each Character.

We can distinguish its vertical, horizontal, arc shapes using a CNN core matrix.

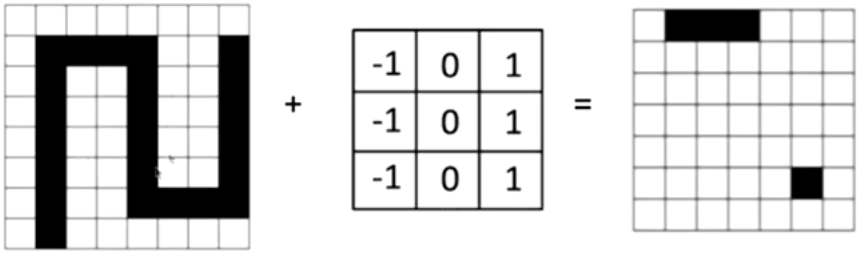

Figure 7: Separating the Horizontal Parameters of the Number.

The nucleus matrix achieves a drastic reduction in the values of access to the neuron by isolating the main parts.

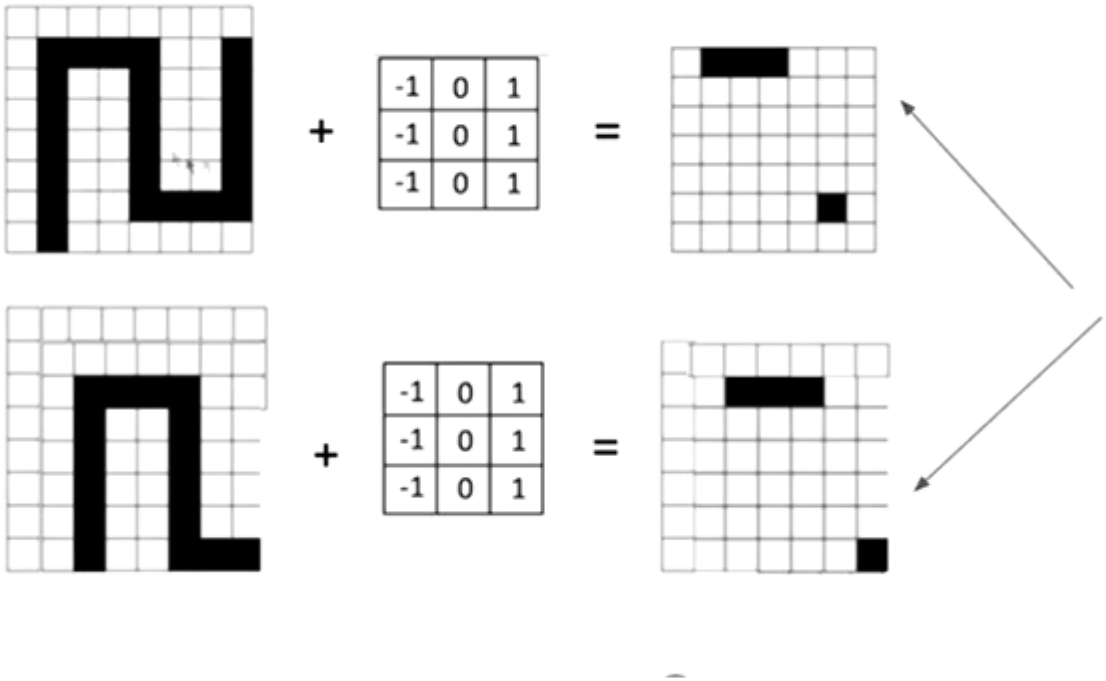

Figure 8: Separation of the Main Features by 2 Core Switches.

A series image is inserted into the CNN input layer.

$$(f * g)[m,n] = \sum_{k,l} f[m-k, n-l] * g[k,l] \quad (1)$$

Here **f** is the input matrix and **g** is the core matrix.

A new active matrix is formed by a kernel (1) of the given nxn size: a pool is created in the active matrix.

The results from the row segmentation separate and classify the glyphs that are the basis for each row, that is, the general forms of word and letter forms.

The Result

```
from tkinter import *
from PIL import ImageTk, Image
from tkinter import filedialog,messagebox
from ttk import Frame, Label
import configparser
import cv2
import tempfile
import numpy as np
import os
```

import subprocess

libraries are used using tensorflov platform and

hdf5 file capabilities were used.

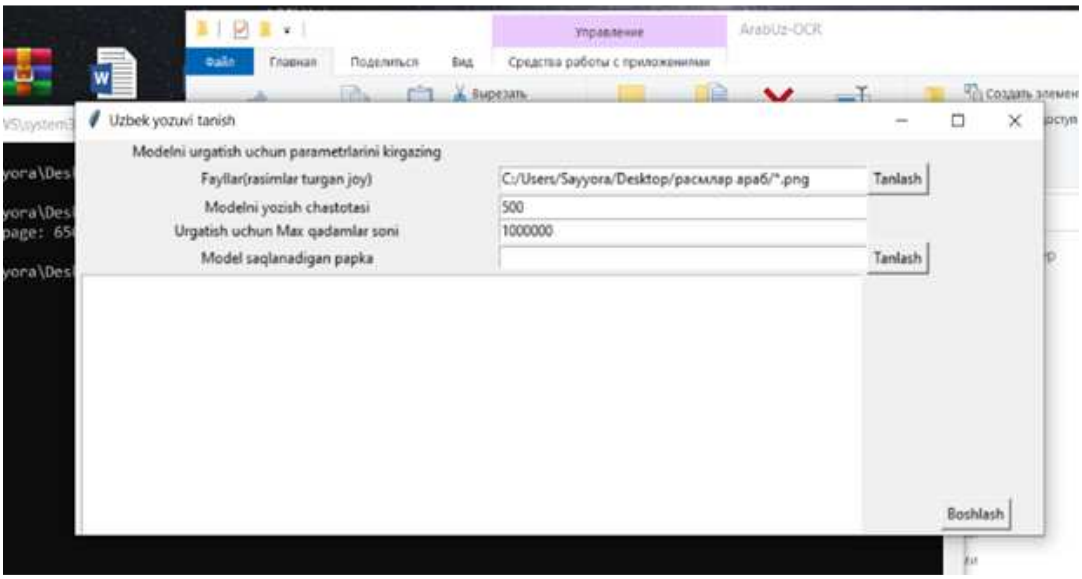

Figure 9. Using Neural Network in Model Training Window.

J is stored in the number extension. With Class Attrs () we can use the functions and procedures available in the library.

```
class Attrs():
    def __init__(self):
        self.files = glob_all([os.path.join(this_dir, "data", "train", "*.png")])
        self.seed = 24
        self.backend = "TensorFlow"
        self.network = "CNN=40:3x3,pool=2x2,CNN=60:3x3,pool=2x2,lstm=200,dropout=0.5"
        self.checkpoint_frequency = 1000
        self.max_iters = 1000
        self.stats_size = 100
        self.no_skip_invalid_gt = False
        self.no_progress_bars = True
        self.output_dir = os.path.join(this_dir, "test_models")
        self.whitelist_files = []
        self.whitelist = []
        self.gradient_clipping_mode = "AUTO"
```

self.n_augmentations = 0

self.fuzzy_ctc_library_path = ""

self.num_inter_threads = 0

self.num_intra_threads = 0

self.text_regularization = ["extended"]

self.text_normalization = "NFC"

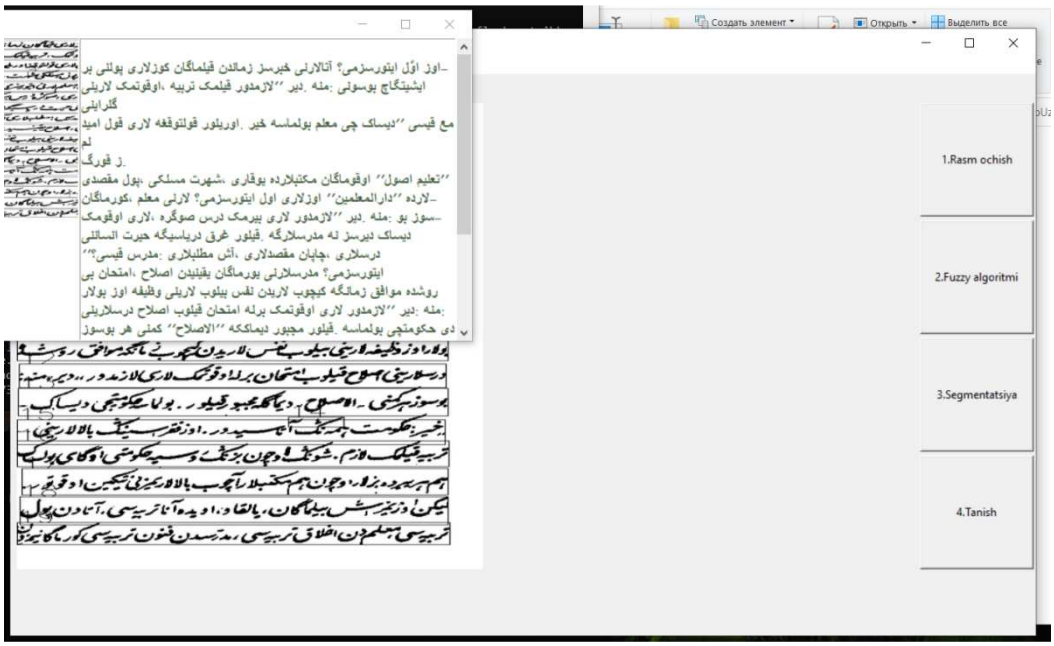

Figure 10. Neural Network Usage Interface in Image Recognition.

Based on the module created by leNet, it is possible to create models by teaching images using ready-made modules to simplify calculations and achieve results. Using the generated model, it is possible to obtain familiar results using the necessary functions in image recognition. Several neural models are available in the TensorFlow library and provide opportunities to perform their comparative analyzes. Open data sets are available at https://www.kaggle.com. They are data sets for letters, words, lines, magazines, books, and with the help of our software we used the following data sets to recognize Arabic text.

Table 1: Results Obtained in the Acquaintance

No.	Data Set	Kitob Nomi	Tanish Foizi
1	Arabic Handwritten Characters Dataset (https://www.kaggle.com/mloey1/ahcd1)	Abdulla Avloniy asari	55%
2	Arabic Handwritten Characters Dataset (https://www.kaggle.com/mloey1/ahcd1)	Muhammadzade Khadizade	60%
3	Urdu Handwritten Characters Dataset https://www.kaggle.com	Abdulla Avloniy asari	62%
4	Urdu Handwritten Characters Dataset https://www.kaggle.com	Muhammadzade Khadizade	60%

We have expanded these data sets for the old Uzbek script. Up to 4,000 sample manuscripts were taken for each letter. There are 28 letters in Arabic and 32 letters in Old Uzbek. We added 4 sides to the data set. That is, we found and placed up to 2,000 samples for each letter.

Table 2: Results from the Extended Data Set

No.	Data Set	Kitob Nomi	Tanish Foizi
1	Olb Uzbek Handwritten Characters Dataset (https://www.kaggle.com/mloey1/ahcd1)	Abdulla Avloniy asari	90%
2	Arabic Handwritten Characters Dataset (https://www.kaggle.com/mloey1/ahcd1)	Muhammadzade Khadizade	87%

CONCLUSIONS AND RECOMMENDATIONS

Results were obtained using a TensorFlow library in a python programming environment to recognize Arabic graphics. TensorFlow library made it possible to facilitate neural network accounting problems. In programming, python has program code close to human language, supports opencv capabilities, allows you to use ready-made libraries using the TensorFlow platform. It is possible to expand a given graph by expanding open-type data sets. The results of recognizing lithographs in Arabic graphics with 90% accuracy were obtained. When data sets are formed separately for each calligrapher in recognition with an accuracy of up to 90 percent, we can achieve efficiency in recognizing his book by selecting the calligrapher as we choose the language in finereadear because each calligraphy has its own style. The software can be used at the Imam Bukhari International Research Center to print more than 70,000 Arabic-language sources in Old Uzbek. The development of neural networks, i.e. the introduction of new models, the introduction of special libraries in the python programming language, the availability of open data sets, These capabilities serve to easily and easily process the problems facing programming and bring accurate results.

LIST OF USED LITERATURE

1. *Yosinski, Jason; Clune, Jeff; Nguyen, Anh; Fuchs, Thomas; Lipson, Hod (2015-06-22). "Understanding Neural Networks Through Deep Visualization". arXiv:1506.06579 [cs.CV].*
2. *"Toronto startup has a faster way to discover effective medicines". The Globe and Mail. Retrieved 2015-11-09.*
3. *"Startup Harnesses Supercomputers to Seek Cures". KQED Future of You. 2015-05-27. Retrieved 2015-11-09.*
4. *Tim Pyrkov; Konstantin Slipensky; Mikhail Barg; Alexey Kondrashin; Boris Zhurov; Alexander Zenin; Mikhail Pyatnitskiy; Leonid Menshikov; Sergei Markov; Peter O. Fedichev (2018). "Extracting biological age from biomedical data via deep learning: too much of a good thing?". Scientific Reports. 8 (1): 5210. Bibcode:2018NatSR...8.5210P. doi:10.1038/s41598-018-23534-9. PMC 5980076. PMID 29581467.*
5. *Chellapilla, K; Fogel, DB (1999). "Evolving neural networks to play checkers without relying on expert knowledge". IEEE Trans Neural Netw. 10 (6): 1382–91. doi:10.1109/72.809083. PMID 18252639.*
6. *Chellapilla, K.; Fogel, D.B. (2001). "Evolving an expert checkers playing program without using human expertise". IEEE Transactions on Evolutionary Computation. 5(4): 422–428. doi:10.1109/4235.942536.*
7. *Fogel, David (2001). Blondie24: Playing at the Edge of AI. San Francisco, CA: Morgan Kaufmann. ISBN 978-1558607835.*

8. *Clark, Christopher; Storkey, Amos (2014). "Teaching Deep Convolutional Neural Networks to Play Go". arXiv:1412.3409 [cs.AI].*
9. *Maddison, Chris J.; Huang, Aja; Sutskever, Ilya; Silver, David (2014). "Move Evaluation in Go Using Deep Convolutional Neural Networks". arXiv:1412.6564[cs.LG].*
10. *"AlphaGo – Google DeepMind". Archived from the original on 30 January 2016. Retrieved 30 January 2016.*
11. *Bai, Shaojie; Kolter, J. Zico; Koltun, Vladlen (2018-04-19). "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". arXiv:1803.01271 [cs.LG].*
12. *Yu, Fisher; Koltun, Vladlen (2016-04-30). "Multi-Scale Context Aggregation by Dilated Convolutions". arXiv:1511.07122 [cs.CV].*
13. *Borovykh, Anastasia; Bohte, Sander; Oosterlee, Cornelis W. (2018-09-17). "Conditional Time Series Forecasting with Convolutional Neural Networks". arXiv:1703.04691 [stat.ML].*
14. *Mittelman, Roni (2015-08-03). "Time-series modeling with undecimated fully convolutional neural networks". arXiv:1508.00317 [stat.ML].*
15. *Chen, Yitian; Kang, Yanfei; Chen, Yixiong; Wang, Zizhuo (2019-06-11). "Probabilistic Forecasting with Temporal Convolutional Neural Network". arXiv:1906.04397 [stat.ML].*
16. *Zhao, Bendong; Lu, Huanzhang; Chen, Shangfeng; Liu, Junliang; Wu, Dongya (2017-02-01). "Convolutional neural networks for time series classi". Journal of Systems Engineering and Electronics. 28 (1): 162–169. doi:10.21629/JSEE.2017.01.18.*
17. *Petneházi, Gábor (2019-08-21). "QCNN: Quantile Convolutional Neural Network". arXiv:1908.07978 [cs.LG].*
18. *Hubert Mara (2019-06-07), HeiCuBeDa Hilprecht – Heidelberg Cuneiform Benchmark Dataset for the Hilprecht Collection (in German), heiDATA – institutional repository for research data of Heidelberg University, doi:10.11588/data/IE8CCN*