АВТОМАТТАНДЫРУ, ТЕЛЕМЕХАНИКА, БАЙЛАНЫС, АҚПАРАТТЫҚ ЖҮЙЕЛЕР
AUTOMATION, TELEMECHANICS, COMMUNICATIONS, POWER ENGINEERING, INFORMATION SYSTEM
АВТОМАТИЗАЦИЯ, ТЕЛЕМЕХАНИКА, СВЯЗЬ, ИНФОРМАЦИОННЫЕ СИСТЕМЫ

УДК 004.85 **DOI 10.52167/1609-1817-2023-126-3-189-197**

В.А. Мадин, О.С. Салыкова, И.В. Иванова
Костанайский региональный университет им. А.Байтурсынова, Костанай, Казахстан
E-mail: vmadin@mail.ru

СВЁРТОЧНЫЕ И ГЛУБОКИЕ НЕЙРОННЫЕ СЕТИ

Аннотация. Подход на основе современных методов машинного обучения делает возможным корректный анализ поведенческих факторов в автоматическом режиме, что является необходимой составной частью современных систем мониторинга, контроля доступа, различных маркетинговых инструментов и т.д. В статье рассмотрены современные подходы к машинному обучению, включая глубокое обучение и тензоризацию искусственных нейронных сетей, предложена разработка нейросетевого классификатора для анализа поведенческих факторов на языке программирования python, приведены результаты построения нейросетевого классификатора для задачи автоматизированной аутентификации пользователей.

Ключевые слова. Искусственная нейронная сеть, нейрон, свёрточная сеть, окно свёртки, тензоризация.

Введение.

Глубокая ИНС (рисунок 1), аналогично зрительной системе человека, при распознавании, например, лица человека на изображении может на первом внутреннем слое производить идентификацию некоторых простых форм (овалов, линий, углов), на следующем слое выявленные сущности могут усложняться (при наличии сигналов от нейронов предыдущего слоя, ответственных, например, за выявление овалов, может приниматься решение о наличии глаза на изображении) и далее от слоя к слою происходит дальнейшее усложнение сущностей, которое в итоге приводит к ответу сети о наличии или отсутствии лица на изображении [1].

Описанная иерархичность обработки входной информации обеспечивает результаты, которые глубокие ИНС демонстрируют в задачах распознавания и классификации [2].

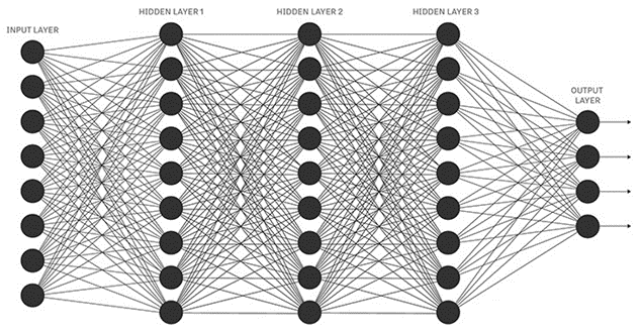


Рисунок 1 – Глубокая нейронная сеть

Важное направление современного развития нейросетевого подхода – это свёрточные ИНС [3], адаптированные для выявления закономерностей и работы с двухмерными и трехмерными входными структурами данных, в частности, для работы с графическими изображениями. В качестве примера на рисунке 2 схематически изображен входной слой свёрточной сети размера 28×28 (соответствующий изображению размера 28×28) и следующий за ним свёрточный слой.

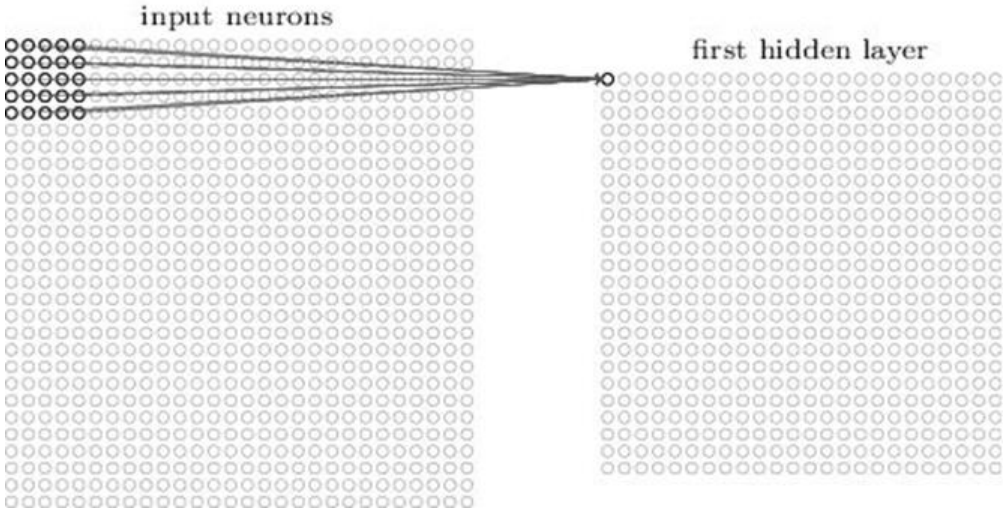


Рисунок 2 – Свертка в рамках сверточных нейронных сетей

Так как один сверточный слой способен выявить только одну особенность, то логичным представляется параллельное использование нескольких сверточных слоев, каждый из которых характеризуется своими весами и смещениями и нацелен на выявление некоторой особенности изображения. В реальных приложениях количество сверточных подслоев в одном слое достаточно большое (рисунок 3), так как их число должно соответствовать количеству важных особенностей изображения.

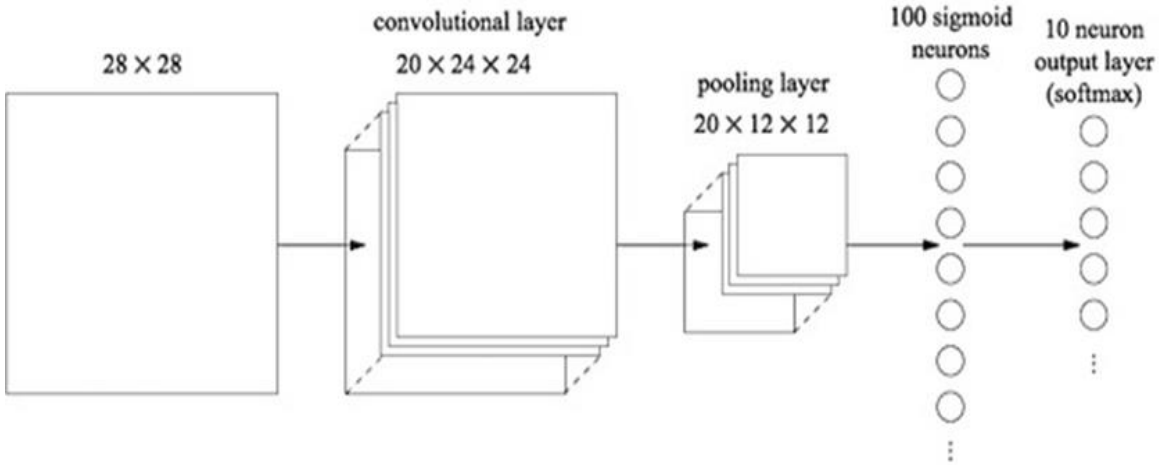


Рисунок 3 – Сверточная нейронная сеть

Обучение регрессионных и нейросетевых моделей является итеративным процессом, при котором параметры модели постепенно подстраиваются с использованием обучающего набора данных. Формально этот процесс можно определить как минимизацию функции потерь $L = L(x(T))$, которая зависит от уровня отклонения прогноза сети от правильных ответов или от скрытых параметров модели. Согласно [4],

классическим способом обучения нейронных сетей является применение стохастического метода градиентного спуска и связанного с ним метода обратного распространения ошибки. При этом на каждой итерации выполняется расчет градиента функционала потерь и изменяются параметры сети в противоположном направлении градиента, обеспечивающем малое изменение [5,6].

Имеющаяся связь между ИНС и обыкновенными дифференциальными уравнениями, занимает важное место в современных исследованиях. Так в [7] демонстрируется возможность перехода от классического рассмотрения ИНС в форме упорядоченных слоев искусственных нейронов к их компактному представлению в виде обыкновенных дифференциальных уравнений.

Материалы и методы.

Для практической реализации алгоритмов функционирования и обучения ИНС, нами была рассмотрена и практически реализована следующая модельная постановка задачи анализа поведенческих факторов и аутентификации пользователей [8]:

1) Предварительное обучение нейросетевого классификатора на размеченном наборе данных MNIST, соответствующим отсканированным рукописным цифрам.

2) Разработка системы трекинга движений указателя мыши при имитации пользователем отрисовки указателем мыши одной из 10 возможных цифр.

3) Проведения распознавания введенной пользователем цифры с использованием обученной ИНС.

Отметим, что данная система создана в иллюстративных целях, однако уже в такой реализации она позволяет успешно решить важную подзадачу, связанную с выявлением автоматизированных систем, имитирующих реальных пользователей. С другой стороны, предложенный подход может использоваться для накопления базы изображений, соответствующих результатам ввода цифр конкретными пользователями, что в дальнейшем позволит проводить надежную аутентификацию пользователей.

Для программной реализации мы использовали язык программирования python и стандартные python модули numpy и matplotlib для проведения векторизованных вычислений и построения графиков соответственно. Обработка изображений осуществлялась с помощью библиотеки pil, а трекинг передвижений мыши – с помощью специализированной библиотеки pynput.

В рамках парадигмы объектно-ориентированного программирования мы разбили программу на отдельные составные части (модули), каждая из которых содержит описание соответствующего класса: act_func – класс, описывающий функцию активации; cost_func – класс, описывающий функцию стоимости; layer – класс, описывающий один слой ИНС, включая механизмы прямого и обратного распространения сигнала; ann – класс, описывающий непосредственно ИНС как упорядоченный набор отдельных слоев; data – класс, предоставляющий набор методов для загрузки и отображения данных (графических изображений с рукописными цифрами); hymabeh – класс, предоставляющий набор методов для запуска и осуществления процедуры трекинга движений указателя мыши.

Результаты.

Запуск программного кода и отображение результатов мы осуществляем с использованием интерактивного браузерного интерфейса jupyter, который активно используется при работе на языке программирования python (в рамках данного интерфейса возможно создание отдельных ячеек, содержащих куски программного кода на языке python, а также ячеек, содержащих отформатированный текст в формате markdown, что делает процесс разработки программного кода и тестовых запусков максимально удобным).

На рисунке 4 производится загрузка всех размеченных изображений рукописных цифр из базы MNIST, и отрисовка одной (случайной) цифры.

```
DT = Data().load_mnist('./../data/mnist/mnist.pkl.gz')
DT.present(50)
Data loaded
Total time =     1.31 sec.
Image: trn #50 | Content: "3"
```

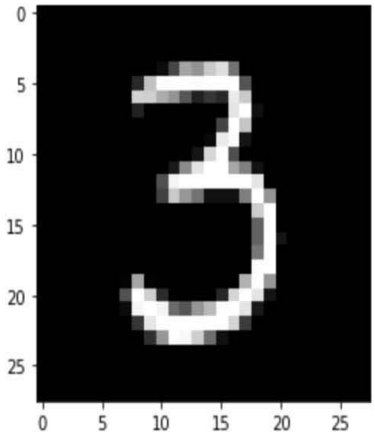


Рисунок 4 – Загрузка изображений рукописных цифр

Далее на рисунке 5 мы приводим скриншот программного кода, соответствующего созданию и обучению ИНС для распознавания рукописных цифр.

```
NET = ANN(mb_size=10, eta=3)
NET.add(LayerFC().init(None, 784))
NET.add(LayerFC().init(784 ,  30))
NET.add(LayerFC().init( 30 ,  10))
NET.prep().learning(DT.x['trn'], DT.y['trn'], DT.x['tst'], DT.y['tst'], epochs=50)
NET.show()

Epoch #  1: duration =     6.55; runs =     50000; err = 0.173600
Epoch #  2: duration =     8.64; runs =    100000; err = 0.158300
Epoch #  3: duration =     8.10; runs =    150000; err = 0.155800
Epoch #  4: duration =     7.27; runs =    200000; err = 0.070000

Epoch # 49: duration =     6.29; runs =   2450000; err = 0.053700
Epoch # 50: duration =     6.26; runs =   2500000; err = 0.053600
```

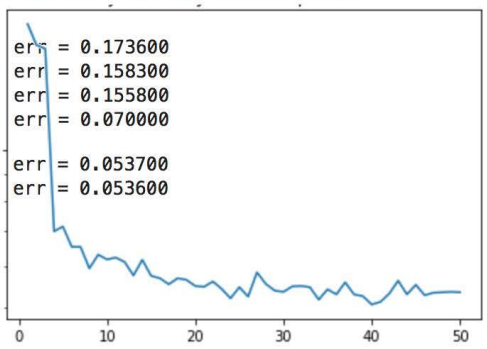


Рисунок 5 – Создание и обучение ИНС

Мы создаем ИНС, содержащий входной слой из 784 нейронов (именно столько пикселей содержат изображения из обучающего набора данных), скрытый слой из 30 нейронов и выходной слой из 10 нейронов (каждый из нейронов выходного слоя соответствует вероятности нахождения на картинке соответствующей цифры). Затем происходит обучение созданной ИНС на 50 эпохах. Как можно видеть на рис. 6, финальная ошибка сети оказывается около 5 процентов, то есть примерно 95 картинок из 100 начинают распознаваться правильно.

Следует отметить, что изменение структуры ИНС (количество слоев, нейронов, замена функции активации и т.д.), а также увеличение или уменьшение эпох обучения, как правило, приводит к изменению результата работы обученной ИНС. Что, в свою очередь, может привести к возникновению «переобучения» ИНС и уменьшению количества верных распознаваний новых наблюдений. Именно поэтому этап формирования архитектуры и обучения современных ИНС относится к наиболее важным и требовательным к вычислительным ресурсам.

На рисунке 6 мы приводим пример корректного результата распознавания случайного изображения из тестового набора данных, а на рисунке 7 демонстрируем пример первой неверно распознанной цифры.

```
NET = ANN.load('./saved/ann_num')
i = 20

x = DT.get(i, 'tst')[0]
y = DT.get(i, 'tst')[1]
a = NET.forward(x)

print('-'*29)
print('Real number is           : ', np.argmax(y))
print('The answer of the ANN is : ', np.argmax(a))
print('-'*29)

DT.present(i, dt='tst')
```

```
-----------------------------
Real number is           :  9
The answer of the ANN is :  9
-----------------------------
Image: tst #20 | Content: "9"
```

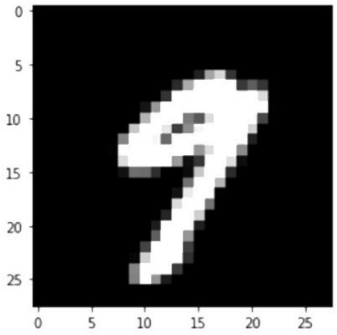


Рисунок 6 – Сохранение/загрузка ИНС и пример предсказания

```
for i in range(DT.n['tst']):

    x = DT.get(i, 'tst')[0]
    y = DT.get(i, 'tst')[1]
    a = NET.forward(x)

    if np.argmax(y) != np.argmax(a):
        break

print('-'*29)
print('Real number is           : ', np.argmax(y))
print('The answer of the ANN is : ', np.argmax(a))
print('-'*29)

DT.present(i, dt='tst')
```

```
-----------------------------
Real number is           :  3
The answer of the ANN is :  8
-----------------------------
Image: tst #18 | Content: "3"
```

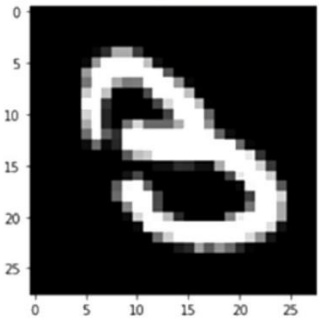


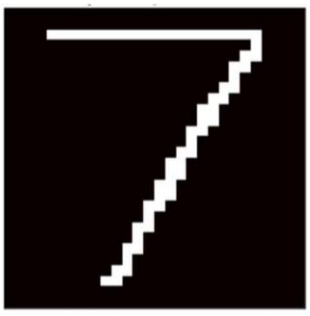
Рисунок 7 – Пример ошибочного предсказания ИНС

Далее мы реализуем систему трекинга действий пользователя. По запросу системы пользователь осуществляет движение указателем мыши, соответствующее указанной цифре. После отпускания клавиши мыши происходит фиксация трека, построение сглаженного изображения и его распознавание обученной ИНС (рисунок 8).

```
HM = Hymabeh()
HM.start()

x = HM.vect()
a = NET.forward(x)

print('-'*29)
print('Real number is           : ', HM.numb)
print('The answer of the ANN is : ', np.argmax(a))
print('-'*29)
```



```
-----------------------------
Real number is           :  7
The answer of the ANN is :  7
-----------------------------
```

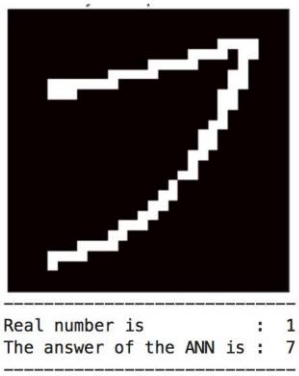
Рисунок 8 – Пример №1 запуска системы трекинга

На рисунке 9 пример неверного результата распознавания (была нарисована не очень удачно цифра 1, которую система распознала как цифру 7).

```
HM = Hymabeh()
HM.start()

x = HM.vect()
a = NET.forward(x)

print('-'*29)
print('Real number is           : ', HM.numb)
print('The answer of the ANN is : ', np.argmax(a))
print('-'*29)
```



```
-----------------------------
Real number is           :  1
The answer of the ANN is :  7
-----------------------------
```

Рисунок 9 – Пример неверного распознавания

Обсуждение.

Искусственные нейронные сети стали необходимыми в приложениях, связанных с машинным зрением, автоматизированным переводом, обработкой видеоданных и других задачах. В ближайшие годы они будут использоваться в автономных системах вождения, роботизированных решениях на производстве и биомедицинских приложениях, однако для практического использования методов нейросетей для развития научно-технического прогресса необходимо развивать новые алгоритмы. Эта работа показывает возможность перехода от классического взгляда на нейронные сети к их компактному представлению в виде обыкновенных дифференциальных уравнений, что ускоряет значительно их обучение.

Для вывода формул обучения ИНС, используем общий случай многослойной структуры и изменим вариант дискретизации Эйлера для модельного уравнения в форме:

$$x(t_{l+1}) = x(t_l) + \Delta t f(x(t_l), q(t_l)), \tag{1}$$

из которой следует:

$$\frac{\partial x(t_{l+1})}{\partial x(t_l)} = I + \Delta t \frac{\partial f(x(t_l), q(t_l))}{\partial x(t_l)}, \tag{2}$$

где I – единичная матрица.

Используя частичное дифференцирование для вывода производной функциональных входных потерь l-ого слоя, запишем:

$$a(t_l) = \frac{\partial L}{\partial x(t_l)} = \frac{\partial L}{\partial x(t_{l+1})} \frac{\partial x(t_{l+1})}{\partial x(t_l)} = a(t_{l+1})(I + \Delta t \frac{\partial f(x(t_l), q(t_l))}{\partial x(t_l)}) \tag{3}$$

и тогда имеем:

$$\frac{a(t_{l+1}) - a(t_l)}{\Delta t} = -a(t_{t+1}) \frac{\partial f(x(t_l), q(t_l))}{\partial x(t_l)}, \tag{4}$$

где введенный вектор $a(t) = \frac{\partial L}{\partial x(t_l)} d$ удовлетворяет дифференциальному уравнению (в пределе $\Delta t \to 0$):

$$\frac{\partial a(t)}{\partial t} = -a(t)\frac{\partial f(x(t), q(t))}{\partial x(t)}. \quad (11)$$

Разработанный формализм может быть применен для изображения классических полносвязных нейронных сетей в виде обыкновенных дифференциальных уравнений, так как функция будет описывать преобразования сигнала каждого слоя сети. Учитывая стремление к бесконечному количеству слоев, где каждый из слоев должен иметь незначительное пропорциональное влияние, необходимо установить для l-ого слоя определенное значение:

$$W_l = +w_{l\square t}, \quad (5)$$

$$B_l = b_l\square t, \sigma_l(\tau) = \tau - \mu\tau^3\square t \quad (6)$$

и тогда полносвязная ИНС может быть представлена в форме уравнения (1) с правой частью:

$$f(x(t), W(t), b(t)) = W(t)x(t) + b(t) - \mu x^3(t). \quad (7)$$

Отметим, что в данном случае роль параметров модели $q(t)$ играют веса нейронов в форме матричной функции $W(t)$ и смещения нейронов в форме векторной функции $b(t)$, которые являются непрерывными функциями времени.

Применяя производную функции по $x(t)$:

$$\frac{\partial f}{\partial x(t)} = W(t) - 3\mu x^2(t)I, \quad (8)$$

мы можем, посредством решения уравнения обратного распространения ошибки вычислить $a(t)$, а затем обновить параметры модели в соответствии с производными:

$$\frac{\partial f_k}{\partial W_{ij}(t)} = \delta_{ki}x_j(t), \quad (9)$$

$$\frac{\partial f_k}{\partial b_j(t)} = \delta_{kj}. \quad (10)$$

Существует два явных плюса, связанных с переходом от ИНС к моделированию обыкновенных дифференциальных уравнений. Во-первых, дифференциальные уравнения имеют развитую и богатую теорию, которая может быть распространена на нейросетевой подход, обеспечивая строгую доказательную базу для алгоритмов машинного обучения. Во-вторых, глубокие ИНС позволяют моделировать предметные области, недоступные для однослойных сетей благодаря иерархическому характеру обучения. Дифференциальные уравнения являются предельным случаем, соответствующим бесконечному числу слоев в сети, что может привести к более высокой предсказательной способности алгоритмов машинного обучения. Малоранговые тензорные аппроксимации могут снизить избыточность модельных коэффициентов в подобном представлении ИНС, исследования которого могут раскрыть все перспективы этого подхода.

Заключение.

Созданная в рамках данной работы программная реализация системы аутентификации на основе интеллектуального анализа действий пользователя на языке

python демонстрирует перспективность использования нейросетевого подхода для реальных систем мониторинга и аутентификации пользователей компьютеров.

Подобные системы в перспективе позволят, в частности, незаметным образом производить идентификацию пользователей различных интернет-сервисов, что внесет качественный скачок в развитие таргетированной рекламы и аналитики на веб-сайтах, поскольку сделает возможным без явной авторизации соотносить пользователя с определенным цифровым (возможно анонимным) портретом и получать доступ к истории его действий в сети Интернет.

ЛИТЕРАТУРА

[1] Ryszard S. Michalski, Jaime G. Carbonell, Tom M. Mitchell. (2013). Machine learning: An artificial intelligence approach, Springer Science & Business Media.

[2] Yann LeCun, Yoshua Bengio, Geoffrey Hinton. (2015). Deep learning, Nature 521(7553): 436-444.

[3] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, Victor Lempitsky. (2014). Speeding-up convolutional neural networks using fine-tuned cp-decomposition, arXiv preprint arXiv:1412.6553.

[4] Salykov B., Salykova O., Ivanova I. (2020). E-cient Training of Deep Neural Networks for Pattern Recognition, Journal of Mathematics, Mechanics and Computer Science 3: 42-48.

[5] Yoshua Bengio, Patrice Simard, Paolo Frasconi. (1994). Learning long-term dependencies with gradient descent is di-cult, IEEE transactions on neural networks 5.2: 157-166.

[6] Martin Zinkevich, Markus Weimer, Alex Smola, Lihong Li. (2010). Parallelized stochastic gradient descent, Advances in neural information processing systems.

[7] Пережогин К.А., Салыкова О.С. (2020). Тензоризация глубоких нейронных сетей, «Әлем таныған Абай»: материалы международной научно-практической конференции студентов и магистрантов – Костанай: Костанайский государственный университет имени А.Байтурсынова: 235-240.

[8] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, Victor Lempitsky. (2014). Speeding-up convolutional neural networks using fine-tuned cp-decomposition, arXiv preprint arXiv:1412.6553.

Владимир Мадин, докторант, А.Байтұрсынова атындағы Қостанай өңірлік университеті, Қостанай, Қазақстан, vmadin@mail.ru

Ольга Салыкова, т.ғ.к., қауымдастырылған профессор, А.Байтұрсынова атындағы Қостанай өңірлік университеті, Қостанай, Қазақстан, solga0603@mail.ru

Ирина Иванова, п.ғ.к., қауымдастырылған профессор, А.Байтұрсынова атындағы Қостанай өңірлік университеті, Қостанай, Қазақстан, valera_irina_69@mail.ru

КОНВОЛЮЦИЯЛЫК, ЖЭНЕ ТЕРЕЦ НЕЙРОНДЫК, ЖЕЛШЕР

Аңдатпа. Машиналық оқытудың заманауи әдістеріне негізделген тәсіл автоматты режимдегі мінез-құлық факторларын дұрыс талдауға мүмкіндік береді, бұл қазіргі заманғы мониторинг, қол жетімділікті бақылау жүйелерінің, әртүрлі маркетингтік құралдардың және т. б. қажетті құрамдас бөлігі болып табылады. Мақалада машиналық оқытудың заманауи тәсілдері, соның ішінде терең оқыту және тензоризация қарастырылған.жасанды нейрондық желілер, Python бағдарламалау тіліндегі мінез-құлық

факторларын талдау үшін нейрондық желілік классификаторды әзірлеу ұсынылады, пайдаланушылардың автоматтандырылған аутентификация тапсырмасы үшін нейрондық желілік классификаторды құру нәтижелері келтірілген.

Түйінді сөздер. Жасанды нейрондық желі, нейрон, конволюциялық желі, конволюция терезесі, тензоризация.

Vladimir Madin, doctoral student, Kostanay Regional University A. Baitursynov, Kazakhstan, Kostanay, vmadin@mail.ru

Olga Salykova, candidate of technical sciences, associate professor, Kostanay Regional University A. Baitursynov, Kazakhstan, Kostanay, solga0603@mail.ru

Irina Ivanova, candidate of pedagogical sciences, associate professor, Kostanay Regional University A. Baitursynov, Kazakhstan, Kostanay, valera_irina_69@mail.ru

CONVULSIVE AND DEEP NEURAL NETWORKS

Abstract. The approach based on modern machine learning methods makes it possible to correctly analyze behavioral factors in automatic mode, which is a necessary part of modern monitoring systems, access control, various marketing tools, etc. The article discusses modern approaches to machine learning, including deep learning and tensorization of artificial neural networks, proposes the development of a neural network classifier for analyzing behavioral factors in the python programming language, and presents the results of constructing a neural network classifier for the task of automated user authentication.

Keywords. Artificial neural network, neuron, convolutional network, convolution window, tensorization.

****