**УДК 378+517.9+004**
**DOI 10.25688/2072-9014.2021.57.3.02**

**И. В. Левченко,**
**Е. С. Левченко,**
**Л. И. Карташова**

# Методика организации практической работы «Создание и реализация компьютерной модели персептрона» в школьном курсе информатики

В статье предложены методические рекомендации по организации практической работы по созданию и реализации компьютерной модели персептрона на языке программирования Python, направленной также и на закрепление учебного материала по применению однослойных нейронных сетей в распознавании образов.

Ключевые слова: методика обучения информатике в школе; искусственный интеллект; нейронная сеть; персептрон; язык программирования Python.

В данной статье описана методика организации выполнения практической работы, посвященной разработке и реализации компьютерной модели персептрона и проводимой в рамках изучения школьниками основ искусственного интеллекта [13]. Знание учебного материала, который был освоен учащимися на занятиях по темам, связанным с моделированием нейронных сетей, изучением основ языка программирования Python, распознаванием образов интеллектуальными системами [8; 10], является необходимым условием для выполнения практической работы по созданию компьютерной модели персептрона, а также ее дальнейшей реализации [7].

Сначала рассмотрим подход к организации практической работы «Создание и реализация компьютерной модели персептрона» [5]. Практическую деятельность учащихся, связанную с созданием однослойной нейронной сети, целесообразно организовать на основе ранее изученных дидактических элементов курса информатики [3; 6], она же послужит и для закрепления учебного материала по применению нейросетей в распознавании образов.

Помимо этого, результаты выполнения данной практической работы целесообразно обсудить с учащимися с точки зрения возможностей использования интеллектуальных систем для решения самых разнообразных задач и перспектив дальнейшего развития технологий искусственного интеллекта и их применения [12]. Тем самым вы проведете повторение ранее изученного материала, но уже с учетом собственного опыта учащихся, их знаний и умений.

При организации практической деятельности учащихся важно следить за соблюдением санитарно-гигиенических норм, касающихся работы учащихся за компьютерами, уделять внимание физкультминуткам и упражнениям для глаз.

Данную практическую работу следует разделить на две части, каждая из которых может быть реализована как минимум в течение одного занятия. Выделяем структурные элементы работы: постановку задачи, разработку информационной модели, разработку алгоритма и реализацию его этапов в виде создания и выполнения программного кода [11]. Практическую работу предлагается выполнять с использованием системы программирования Python и другого свободно распространяемого программного обеспечения [1; 4; 9].

*Цель занятий:* закрепить знания и выработать умения по созданию прототипа персептрона.

*Тип занятий:* закрепление и развитие знаний и умений.

*Место занятий в курсе:* занятия (как минимум два) позволяют получить практический опыт по созданию систем распознавания естественного языка и являются завершающими в тематическом разделе.

*Обсуждаемыми вопросами* являются ключевые моменты ранее изученного материала тематического раздела.

Среди *практических заданий* присутствует выполнение как основных, так и дополнительных заданий.

### Методические рекомендации

***Первое,*** приступаем к постановке задачи. Необходимо совместно с учащимися сформулировать задачу или же предложить им готовый вариант постановки задачи. В обоих случаях необходимо удостовериться в том, что учащиеся задачу поняли, в условии задачи для них не осталось неясных моментов и они готовы приступить к работе над ней. Приведем пример постановки задачи: «Необходимо разработать компьютерную программу, которая на основе однослойной нейронной сети (персептрона) может распознавать букву О среди букв Б, Г, Е, Н, О, П, Р, С, Т».

***Второе,*** после формулировки задачи, переходим к обсуждению условия задачи с учащимися. Для этого используем систему наводящих вопросов, отвечая на которые учащиеся уже точно смогут разобраться с условием задачи, выделить ее основные компоненты, такие как входящие данные, выходные данные и т. д. Все это позволит учащимся понять и принять задачу, которую необходимо решить, а учителю определить тех, кто задачу не понял и кому необходима дополнительная помощь.

Анализируя таким образом условие задачи, приходим к совместному с учащимися выводу о том, что в качестве входных данных предлагается 9 картинок,

а именно 9 растровых изображений указанных в условии букв, а на выходе должно быть нахождение картинки с буквой О [2]. Обращаем внимание учащихся, что программа вычисляет значения весовых коэффициентов для персептрона с целью распознавания буквы О.

Обсуждаем, что на первом этапе не ставится задача обучить нейросеть, а надо лишь выполнить расчет суммарных значений входных сигналов нейрона для каждой предъявляемой буквы, и здесь имеют место два случая: либо весовые коэффициенты являются случайными, либо, наоборот, они правильно подобраны. Затем обращаем внимание, что полученные суммарные значения сравниваются с порогом чувствительности нейрона и на основании этого делается вывод о нахождении буквы О.

Также следует обсудить с учащимися, что на первом этапе не ставится задача разработать пользовательский графический интерфейс программы. Реализация разрабатываемой программы предполагается с использованием лишь интерфейса командной строки, то есть консольного взаимодействия с компьютером. Отличия графического и командного (или консольного) пользовательского интерфейсов должны быть известны учащимся из общеобразовательного курса информатики.

В дальнейшем данная задача может быть усложнена требованием по созданию пользовательского интерфейса. Такой интерфейс может быть либо изначально определен учителем, либо продуман и разработан учащимися полностью самостоятельно, но с обязательным наличием элементов, указанных на этапе постановки задачи, например таких: растровые изображения 9 букв, над которыми указаны суммы значений входных сигналов и ответ на вопрос задачи в виде слов True или False; таблицы весов; значение порога чувствительности нейрона; поле для ввода порядкового номера проверяемой буквы; сумма значений входных сигналов; ответ на вопрос задачи [11].

***Третье,*** соблюдая логику проведения практической работы, в качестве следующего шага необходимо совместно с учащимися перейти к разработке информационной модели. С помощью системы наводящих вопросов учителя учащиеся приходят к выводу, что изображение каждой из предлагаемых 9 букв можно представить в виде растра 3 × 5 пикселей. Например, растровое черно-белое изображение буквы Б можно записать как последовательность нулей и единиц, где единица — это пиксель черного цвета, ноль — белого. В итоге получим 15 таких значений:

$$[[1,1,1],[1,0,0],[1,1,1],[1,0,1],[1,1,1]].$$

Далее вместе с учащимися обсуждаем, что от каждого пикселя растрового изображения нейрон или получает сигнал ($x = 1$), или не получает его ($x = 0$). Вспоминаем, что каждая связь из имеющихся 15 (так называемые входы в нейрон) имеет свой весовой коэффициент (например, $w = –1$), и все эти коэффициенты хранятся в таблице межнейронных связей. Затем обращаем внимание, что искусственный нейрон суммирует значения входных сигналов $x_j$, которые

умножены на весовые коэффициенты (значения весов) $w_j$. Другими словами, вычисления происходят следующим образом:

$$S = x_1 \cdot w_1 + x_2 \cdot w_{12} + \ldots + x_{15} \cdot w_{15}.$$

Затем уточняем, что искусственный нейрон сравнивает результат суммарного значения входных сигналов $S$ с порогом чувствительности θ и вырабатывает значение выходного сигнала $y$. Значение сигнала у может быть равно 1 (или, другими словами, сигнал есть) в том случае, когда $S$ больше либо равно θ, и равно 0 (сигнала нет) в том случае, когда $S$ меньше θ.

Далее сообщаем, что перед нейрокомпьютером стоит задача выдавать значение $y = 1$ (True) при предъявлении карточки с изображением буквы О и выдавать значение $y = 0$ (False) для любой другой из восьми возможных букв. Предлагаем сначала весовые коэффициенты $w_j$ задать датчиком случайных чисел в диапазоне [–1, 1], а порог чувствительности θ установить принудительно (например, 10). Обращаем внимание, что порог чувствительности выбирается с учетом необходимой точности работы нейрона и допустимой длительности его обучения. Далее обучение нейрокомпьютера идет с помощью изменений весовых коэффициентов (значения весов) $w_j$.

Рассматриваем пример предъявления карточки с буквой Б на входе персептрона. Если случайно выходной сигнал у оказался равен нулю (суммарное значение входных сигналов не превысило порог чувствительности нейрона), то это означает, что корректировать весовой коэффициент не надо, поскольку реакция персептрона верная (на карточке не буква О). А если случайно выходной сигнал у оказался равен единице (на карточке буква О), что неправильно, то следует уменьшить («наказать») весовые коэффициенты тех активных входов (например, $x_2 = 1$), которые способствовали возбуждению нейрона ($w_2$ и др.).

В качестве другого примера рассматриваем предъявление карточки с буквой О на входе персептрона. Если случайно выходной сигнал у оказался равен единице (суммарное значение входных сигналов превысило порог чувствительности нейрона), то это означает, что корректировать весовой коэффициент не надо, поскольку реакция персептрона верная (на карточке буква О). А если случайно выходной сигнал у оказался равен нулю, а это означает, что на карточке не буква О, что неправильно, то следует увеличить («поощрить») весовые коэффициенты тех активных входов (например, $x_2 = 1$), которые способствовали возбуждению нейрона ($w_2$ и др.).

Можно предложить учащимся самостоятельно объяснить обучение персептрона распознаванию буквы О с использованием других букв по аналогии с уже рассмотренной буквой Б. Для упрощения взаимодействия с программой предлагаем учащимся записать блоки, позволяющие вывести на экран изображения предъявляемых картинок, таблицы весов (исходные и текущие значения) межнейронных связей.

***Четвертое,*** переходим к разработке алгоритма на языке программирования Python. Обращаем внимание учащихся, что код программы надо будет

поэтапно пополнять, сохранять и запускать на исполнение, каждый раз верифицируя результат.

Первый этап — это запуск среды разработки и создание файла *pers_1*, что делается с помощью команды меню (*File / New File*). Поскольку предполагается, что значения таблицы исходных весов межнейронных связей будут заполняться случайным образом, то потребуется функция генерации случайных чисел. Для этого будем использовать функцию из внешней библиотеки `random`, которая подключается с помощью инструкции `import random`.

Второй этап связан с формированием и выводом на экран изображений 9 отдельных букв. Каждая картинка с буквой будет представлена в виде *списка*, состоящего, в свою очередь, из *вложенных списков*. Каждый вложенный список — это отдельная строка из 3 элементов. Для каждой картинки будет 5 таких строк.

Затем вспоминаем с учащимися формат записи списка (т. е. упорядоченной последовательности) на языке программирования Python и особенности записи вложенных списков (что аналогично многомерным массивам). Например, первая буква в списке — буква Б — будет записана следующим образом:

```
one = [[1,1,1],[1,0,0],[1,1,1],[1,0,1],[1,1,1]]
```

Таким же образом следует представить и остальные картинки с буквами. Обращаем внимание учащихся, что необходимо сохранить фрагмент кода программы в файле, имя которого уже задано.

Предлагаем вывод изображения отдельной картинки выполнить с помощью функции `pr_number (N)`. Вспоминаем, что отдельные части программы, выполняемые больше одного раза, на языке Python оформляются в виде функции, определение которой начинается со служебного слова `def`, затем записывается имя функции и в круглых скобках указываются формальные параметры, а заканчивается запись двоеточием. В последующих строках записываются инструкции функции с обязательным отступом в 4 пробела в каждой строке. Вызов функции осуществляется по ее имени с указанием фактических параметров в круглых скобках.

Обсуждаем с учащимися, что функция `pr_number (N)` перебирает каждый элемент отдельной картинки по строкам и столбцам (5 строк и 3 столбца) в двух вложенных инструкциях `for` (т. е. циклах с параметром), и в зависимости от записанного значения в отдельном элементе списка `N` (первый из списка параметров функции) либо рисуется символ «`*`», либо нет. Обращаем внимание учащихся, что в текст фрагмента программы для лучшего его понимания включены комментарии, начинающиеся с символа # (шарп).

В результате фрагмент кода программы, описывающий процедуру вывода изображения отдельной картинки, будет таким, как показано на рисунке 1.

Уточняем, что код вызова функции, позволяющей вывести на экран, например изображение первой буквы Б в виде звездочек, будет следующим: `pr_number (one)`. Обсуждаем с учащимися, что необходимо добавить

```
--------------- печать буквы ----------------------
def pr_number(N):
    for i in range(0, 5):
        for j in range(0, 3):
            if N[i][j]==1:
                print("*",end=' ')
            else:
                print(" ",end=' ')
        print()
```

**Рис. 1.** Код вывода изображения буквы

в программу инструкции вывода всех 9 букв, разделив их между собой горизонтальными линиями с помощью инструкции `print` («`----------`»).

Напоминаем учащимся о необходимости сохранения фрагмента кода программы в файле, имя которого уже задано, а также запуска файла на исполнение для верификации результата.

Третий этап связан с генерацией и выводом таблицы исходных весов межнейронных связей. Обсуждаем, что сначала надо заполнить таблицу весов нулевыми значениями, записав элементы таблицы в виде списка:

```
wes = [[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]]
```

Затем таблицу исходных весов межнейронных связей необходимо заполнить случайными целыми числами, находящимися в диапазоне [–1, 1]. Это можно сделать с помощью функции, генерирующей целые числа:

```
random.randint (-1,1)
```

Далее предлагаем учащимся выполнить генерацию элементов списка (т. е. заполнение случайными целыми числами таблицы весов) с помощью функции `create_SYN (SY)`. В результате фрагмент кода программы, описывающий процедуру заполнения случайными значениями таблицы весов, будет таким, как показано на рисунке 2.

```
#------------ создание таблицы случайных весов ------------------------
def create_SYN(SY):      # задание случайных значений таблицы весов SY
    for i in range(0, 5):
        for j in range(0, 3):
            SY[i][j] = random.randint(-1,1)
```

**Рис. 2.** Создание таблицы случайных весов

Далее вызываем функцию `create_SYN`, указав в качестве параметра элементы списка `wes` для заполнения этого списка (таблицы весов) случайными числами: `create_SYN(wes)`. Затем запишем функцию вывода таблицы весов с помощью двух вложенных инструкций `for` (т. е. циклов с параметром) (см. рис. 3).

После этого вызываем процедуру `print_mass`, указав в качестве параметра элементы списка wes для вывода этого списка (таблицы весов), заполненного случайными числами, т. е. `print_mass (wes)`. Сохраняем и запускаем измененный файл, проверяем полученные результаты.

```
#----------------- печать таблицы ---------------------
def print_mass(N):
    for i in range(0, 5):
        for j in range(0, 3):
            print("%.0f" %(N[i][j]),"  ",end=' ')
        print()
```

**Рис. 3.** Вывод таблицы весов

Закончив формирование изображений букв и таблицы весов, можно переходить ко второй части практической работы, которая направлена на формирование таблицы весов с правильными коэффициентами, а значит, на обучение нейросети.

Четвертый этап связан с вычислением суммарного значения входных сигналов нейрона на основе случайных весовых коэффициентов. Вспоминаем с учащимися информационную модель искусственного нейрона, математическую модель нахождения искусственным нейроном суммарного значения входных сигналов, математическую модель описания правила формирования искусственным нейроном выходного сигнала.

На основе ранее рассмотренной схемы персептрона обсуждаем с учащимися алгоритм корректировки весовых коэффициентов. Уточняем, что в процессе обучения персептрона происходит корректировка значений отдельных весовых коэффициентов. Обращаем внимание учащихся, что в результате такого обучения получается таблица весов межнейронных связей, которая позволяет распознавать заданную букву. Еще раз делаем акцент на том, что решаем задачу по распознаванию только буквы О, т. е. работает формула «да/нет» или «она/не она».

Обращаем внимание учащихся, что суммарное значение входных сигналов от каждого пикселя растрового изображения представляет собой сумму произведений значений элементов списка, хранящих изображение букв, на соответствующие элементы таблицы весов. Сообщаем, что для получения таблицы, состоящей из произведений сигналов на коэффициенты веса, будем использовать функцию `mul_SXW(XX,WW,MU)` (рис. 4).

```
def mul_SXW(XX,WW,MU): # перемножение  XX WW (сигналы на веса)  получается MU
    for i in range(0, 5):
        for j in range(0, 3):
            MU[i][j] = XX[i][j] * WW[i][j]
```

**Рис. 4.** Произведение сигналов на коэффициенты веса

Затем говорим о том, что суммарное значение полученных произведений (значений входных сигналов) вычисляется с помощью функции `out_sig(massiv)` (рис. 5).

Обсуждаем, что в функциях используются две вложенные инструкции `for` (т. е. цикл с параметром). В качестве фактического параметра в данную функцию будет передаваться массив произведений сигналов на соответствующие

```
def out_sig(massiv):
    signal = 0
    for i in range(0, 5):
        for j in range(0, 3):
            signal = signal + massiv[i][j]
    return signal
```

**Рис. 5.** Суммарное значение входных сигналов

им веса. Уточняем, что такой способ организации данных, как массив, на языке программирования Python реализуется с помощью списка.

Фрагмент кода программы, позволяющей вывести суммарное значение входных сигналов, рассчитанных на основе случайных весовых коэффициентов, например от первой буквы, будет таким, как показано на рисунке 6.

```
sig   = [[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]]
print("-----суммарное значение входных сигналов--------")
mul_SXW(one,wesp,sig)
print(out_sig(sig), " от первой буквы ")
```

**Рис. 6.** Суммарное значение входных сигналов от первой буквы

Далее обращаем внимание учащихся, что необходимо добавить в программу инструкции вывода суммарных значений входных сигналов для всех 9 букв. Затем следует сохранить фрагмент кода программы в файле, имя которого уже задано, и запустить его на исполнение.

После этого обсуждаем с учащимися результаты работы программы, а именно: полученные на основе случайных весовых коэффициентов суммарные значения входных сигналов нейрона для каждой буквы сравниваем с порогом чувствительности нейрона. Вместе с учащимися убеждаемся, что распознавание буквы О не выполнено, т. е. не получена ситуация, когда от всех букв, кроме пятой буквы, значения стали меньше 10.

Пятый этап заключается в выводе таблицы весов с правильными коэффициентами. Для этого необходимо в соответствии с правильно подобранными весовыми коэффициентами для распознавания буквы О заполнить значения элементов списка wesp:

```
wesp = [[2,0,0],[0,1,1],[1,-1,2],[0,-1,0],[2,3,0]]
```

Далее вызываем функцию `print_mass`, указав в качестве параметра элементы списка `wesp` для вывода этого списка (таблицы весов), заполненного случайными числами: `print_mass(wesp)`. Напоминаем о необходимости сохранить файл и проверить работу программы.

Шестой этап заключается в том, что на нем происходит вычисление для всех букв суммарных значений входных сигналов нейрона на основе правильных весовых коэффициентов. Фрагмент кода программы, позволяющей вывести суммарное значение входных сигналов, рассчитанных на основе правильных весовых коэффициентов, взятых, например, для первой буквы, будет таким, как показано на рисунке 7.

```
print("-----суммарное значение входных сигналов--------")
mul_SXW(one,wesp,sig)
print(out_sig(sig), " от первой буквы ")
```

**Рис. 7.** Суммарное значение входных сигналов от первой буквы

Далее сообщаем учащимся о необходимости добавить в программу инструкции вывода суммарных значений входных сигналов для всех 9 букв, а затем выполнить верификацию результата, предварительно сохранив изменения.

Седьмой этап связан с распознаванием буквы при сравнении значений с порогом чувствительности нейрона. Уточняем, что для этого необходимо полученные суммарные значения входных сигналов нейрона на основе правильных весовых коэффициентов для каждой буквы сравнить с порогом чувствительности нейрона (установлен равным 10). Вместе с учащимися убеждаемся, что только от пятой буквы значение будет не меньше этого порога (11 >= 10). Делаем вывод о выполнении распознавания буквы О.

Восьмой этап — это внесение изменений в код программы, что не является обязательным. Такое дополнительное задание позволит лучше понять возможности обучения однослойной нейросети по распознаванию растрового изображения. На этом этапе можно изменить порог чувствительности нейрона и выяснить влияние этого параметра на результат распознавания. Также вместо букв можно распознавать цифры или другие символы, внеся изменения в код программы. В конце сохраняем файл и проверяем результат работы программы.

В зависимости от резерва учебного времени и уровня возможностей учащихся, их подготовки учитель может предложить к работе как готовые блоки кода программ, которые необходимо изменять, дополнять и собирать в единое целое, так и дать задание на самостоятельную разработку фрагментов программы.

## Литература

1. Карташова Л. И., Левченко И. В. Методика обучения информационным технологиям учащихся основной школы в условиях фундаментализации образования // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2014. № 2 (28). С. 25–33.

2. Карташова Л. И., Левченко И. В., Павлова А. Е. Обучение учащихся основной школы технологии работы с графическими изображениями, инвариантное относительно программных средств // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2014. № 1 (27). С. 37–46.

3. Кузнецов А. А. Содержание обучения информатике в основной школе: на пути к фундаментализации / А. А. Кузнецов и др. // Вестник Российского университета дружбы народов. Серия «Информатизация образования». 2010. № 4. С. 5–17.

4. Левченко И. В. Информационные технологии в общеобразовательном курсе информатики в контексте фундаментализации образования // Вестник Российского университета дружбы народов. Серия «Информатизация образования». 2018. Т. 15. № 3. С. 282–293.

5. Левченко И. В. Основные подходы к обучению элементам искусственного интеллекта в школьном курсе информатики // Информатика и образование. 2019. № 6. С. 7–15.

6. Левченко И. В. Формирование инвариантного содержания школьного курса информатики как элемента фундаментальной методической подготовки учителей информатики // Вестник Российского университета дружбы народов. Серия «Информатизация образования». 2009. № 3. С. 61–64.

7. Левченко И. В., Абушкин Д. Б., Карташова Л. И. Модуль «Машинное обучение систем искусственного интеллекта» в общеобразовательном курсе информатики // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2020. № 4 (54). С. 27–38.

8. Левченко И. В., Абушкин Д. Б., Михайлюк А. А. Модуль «Восходящее моделирование интеллектуальной деятельности» в общеобразовательном курсе информатики // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2020. № 4 (54). С. 39–50.

9. Левченко И. В., Карташова Л. И., Павлова А. Е. Обучение информационным технологиям в условиях информатизации образования. Воронеж: Научная книга, 2016. 131 с.

10. Левченко И. В., Карташова Л. И., Тамошина Н. Д. Модуль «Нисходящее моделирование интеллектуальной деятельности» в общеобразовательном курсе информатики // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2020. № 3 (53). С. 26–39.

11. Левченко И. В., Левченко Е. С., Михайлюк А. А. Практические работы элективного курса «Основы искусственного интеллекта». М.: Образование и Информатика, 2019. 64 с.

12. Левченко И. В., Павлова А. Е., Садыкова А. Р. Модуль «Введение в искусственный интеллект» в общеобразовательном курсе информатики // Вестник Московского городского педагогического университета. Серия «Информатика и информатизация образования». 2020. № 3 (53). С. 40–51.

13. Левченко И. В. Элективный курс «Основы искусственного интеллекта» / И. В. Левченко и др. М.: Образование и Информатика, 2019. 96 с.

## Literatura

1. Kartashova L. I., Levchenko I. V. Metodika obucheniya informacionny`m texnologiyam uchashhixsya osnovnoj shkoly` v usloviyax fundamentalizacii obrazovaniya // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2014. № 2 (28). S. 25–33.

2. Kartashova L. I., Levchenko I. V., Pavlova A. E. Obuchenie uchashhixsya osnovnoj shkoly` texnologii raboty` s graficheskimi izobrazheniyami, invariantnoe otnositel`no programmny`x sredstv // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2014. № 1 (27). S. 37–46.

3. Kuzneczov A. A. Soderzhanie obucheniya informatike v osnovnoj shkole: na puti k fundamentalizacii / A. A. Kuzneczov i dr. // Vestnik Rossijskogo universiteta druzhby` narodov. Seriya «Informatizaciya obrazovaniya». 2010. № 4. S. 5–17.

4. Levchenko I. V. Informacionny`e texnologii v obshheobrazovatel`nom kurse informatiki v kontekste fundamentalizacii obrazovaniya // Vestnik Rossijskogo universiteta druzhby` narodov. Seriya «Informatizaciya obrazovaniya». 2018. T. 15. № 3. S. 282–293.

5. Levchenko I. V. Osnovny`e podxody` k obucheniyu e`lementam iskusstvennogo intellekta v shkol`nom kurse informatiki // Informatika i obrazovanie. 2019. № 6. S. 7–15.

6. Levchenko I. V. Formirovanie invariantnogo soderzhaniya shkol`nogo kursa informatiki kak e`lementa fundamental`noj metodicheskoj podgotovki uchitelej informatiki // Vestnik Rossijskogo universiteta druzhby` narodov. Seriya «Informatizaciya obrazovaniya». 2009. № 3. S. 61–64.

7. Levchenko I. V., Abushkin D. B., Kartashova L. I. Modul` «Mashinnoe obuchenie sistem iskusstvennogo intellekta» v obshheobrazovatel`nom kurse informatiki // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2020. № 4 (54). S. 27–38.

8. Levchenko I. V., Abushkin D. B., Mixajlyuk A. A. Modul` «Vosxodyashhee modelirovanie intellektual`noj deyatel`nosti» v obshheobrazovatel`nom kurse informatiki // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2020. № 4 (54). S. 39–50.

9. Levchenko I. V., Kartashova L. I., Pavlova A. E. Obuchenie informacionny`m texnologiyam v usloviyax informatizacii obrazovaniya. Voronezh: Nauchnaya kniga, 2016. 131 s.

10. Levchenko I. V., Kartashova L. I., Tamoshina N. D. Modul` «Nisxodyashhee modelirovanie intellektual`noj deyatel`nosti» v obshheobrazovatel`nom kurse informatiki // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2020. № 3 (53). S. 26–39.

11. Levchenko I. V., Levchenko E. S., Mixajlyuk A. A. Prakticheskie raboty` e`lektivnogo kursa «Osnovy` iskusstvennogo intellekta». M.: Obrazovanie i Informatika, 2019. 64 s.

12. Levchenko I. V., Pavlova A. E., Sady`kova A. R. Modul` «Vvedenie v iskusstvenny`j intellekt» v obshheobrazovatel`nom kurse informatiki // Vestnik Moskovskogo gorodskogo pedagogicheskogo universiteta. Seriya «Informatika i informatizaciya obrazovaniya». 2020. № 3 (53). S. 40–51.

13. Levchenko I. V. E`lektivny`j kurs «Osnovy` iskusstvennogo intellekta» / I. V. Levchenko i dr. M.: Obrazovanie i Informatika, 2019. 96 s.

**I. V. Levchenko,**
**E. S. Levchenko,**
**L. I. Kartashova**

**Methodic for Organization Practical Work «Development and Implementation of a Computer Model of the Perceptron» in School Informatic Course**

The article offers methodological recommendations for organizing practical work on the creation and implementation of a computer model of the perceptron in the Python programming language and aimed at consolidating educational material on the use of single-layer neural networks in pattern recognition.

Keywords: methodic of teaching informatic at school; Artificial Intelligence; neural network; perceptron; Python programming language.