ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

УДК 004.032.26

КРАТКАЯ ХАРАКТЕРИСТИКА НЕЙРОННЫХ СЕТЕЙ. РЕАЛИЗАЦИЯ РАСШИРЯЕМОЙ НЕЙРОННОЙ СЕТИ

П.П Боженко, Undeading@yandex.ru, ***Р.У. Стативко***, stativko1@mail.ru
Белгородский государственный технологический университет
(БГТУ) имени В.Г. Шухова, г. Белгород

В данной статье рассмотрено использование нейронных сетей, дана их краткая характеристика. Произведен анализ достоинств, а также существующих на данном этапе становления сложностей, при обучении нейронных сетей на примере предметной области приема на работу. Рассмотрен и описан способ обучения нейронной сети методом обратного распространения ошибки. Выполнена понятная и последовательная реализация универсальной нейронной сети на языке программирования java. Показан пример использования написанной нейронной сети. Приведены и обоснованы результаты обучения написанной нейронной сети, в частности степень изменения весов до начала обучения и после.

Ключевые слова: нейронные сети, программирование нейронных сетей, обучение нейронных сетей, метод обратного распространения ошибки, искусственный интеллект, технологии будущего.

BRIEF CHARACTERISTICS OF NEURAL NETWORKS. REALIZATION OF THE EXPANDABLE NEURAL NETWORK.

P.P. Bozhenko, R.U. Stativko
Shukhov state technological university (BSTU), Belgorod

In the article the use of neural networks is considered, their brief characteristic is given. The analysis of advantages, and also existing at this stage of formation of difficulties, at training of neural networks on the example of a subject area of employment is made. The method of neural network training by the method of error back propagation is considered and described. A clear and consistent implementation of the universal neural network in the java programming language is performed. An example of using a written neural network is shown. The results of training written neural network, in particular the degree of change of weights before and after training are given and justified.

Keywords: neural networks, programming of neural networks, training of neural networks, method of back propagation of errors, artificial intelligence, technologies of the future.

Введение. Внедрение искусственного интеллекта наблюдается в различных отраслях, таких как экономика, медицина и более прикладные области. Ведется большое количество разработок, например в яндексе ведется разработка искусственного интеллекта (ИИ), который может управлять машиной вместо человека, причем испытания проводятся и в условиях города, где ИИ хорошо себя показывает [1]. Возникающая шумиха вокруг искусственного интеллекта в настоящее время преувеличена, однако искусственный интеллект только начал проникать в нашу жизнь, и до конца не понятно, какими большими возможностями и перспективами обладает данная технология.

В этой статье рассмотрены этапы создания и обучения нейронной сети методом обратного распространения ошибок. Также приведен анализ достоинств нейронных сетей и возможных проблем, например, возникающих при их работе в предметной области приема на работу.

ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

Краткая характеристика нейронных сетей. Согласно данным издания Hightech, использование современных алгоритмов и чат ботов позволяет многим предприятиям достаточно сильно сократить время, необходимое на подбор персонала. Исходя из данных отчета Global Recruiting Trends, который был разработан компанией LinkedIn, работодатели наиболее часто сталкиваются с такими проблемами как, сильное разнообразие рынка труда и проведение эффективных собеседований. Для сокращения возможных проблем с поиском нужных сотрудников, многие из этих компаний в серьез думают воспользоваться ИИ, в частности, в частности применяя нейронные сети (НС) [2,3].

Несмотря на то, что НС имеют очень много преимуществ, они также не обделены недостатками, особенно на современном этапе их становления, со времени активного внедрения технологии прошло не так много времени. Так как НС представляет собой некоторую «черную коробку» на вход которой подаются данные, а на выходе получается результат, который невозможно точно определить, ведь он строиться на результате ранее пройденных этапов обучения. Так же не маловажным является то, что при больших объемах данных необходимых для адекватного обучения нейронной сети, сам процесс обучения и «выводы» возникающие у нейронной сети контролировать становиться очень затруднительно, а в некоторых случаях даже невозможно. Примером упомянутых выше тезисов является проблема, возникшая у НС Amazon, в результате которой, НС использующаяся при рассмотрении вакансий приема на работу отдавала предпочтение мужчинам. Однако выяснили это специалисты Amazon не сразу, а после обнаружения данной проблемы быстро отстранили НС от работы. Очевидно, в проектирование системы данное поведение не вкладывали, оно образовалось как раз-таки при обучении системы. Причина почему в процессе обучения НС сделала такие выводы оказалась очень проста, в качестве обучения разработчики использовали базу реальных резюме соискателей, которые в результате всех собеседований были приняты на работу и эффективно справлялись с ней. Большая часть этих резюме было написано именно мужчинами, в результате чего и возник такой механизм у нейронной сети. Специалисты Amazon после данного случая конечно же скорректировали работу НС, убрав данные закономерности, а затем в скором времени «официально» закрыли проект, ведь никто не может гарантировать то, что НС в процессе обучения опять не вернет данный механизм принятия решений.

Представление разработки нейронной сети. Для того, чтобы понять каким образом у НС происходит процесс обучения и работы составлен пример построения двухслойной НС.

Обозначение y1 = значение функции активации нейрона, которая может либо усиливать сигнал, либо наоборот ослаблять. F1 в свою очередь представляет функцию активации, есть множество разных видов функций, но выбор оптимальный не входит в задачи данной статьи. W(x) – вес связи нейрона, в начале задается случайными величинами обычно от -0,5 до 0,5.

Обучением нейронной сети как раз является назначение правильных весов связи. Есть множество разных методов, но в данной статье для примера будет рассматриваться обучение методом обратного распространения ошибки [4].

ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

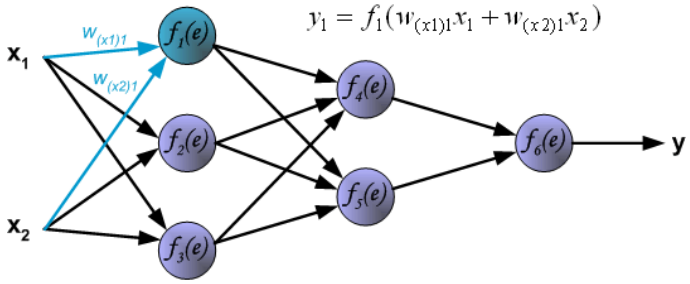

Рис. 1. Пример расчета функции активации нейронов 1-го слоя

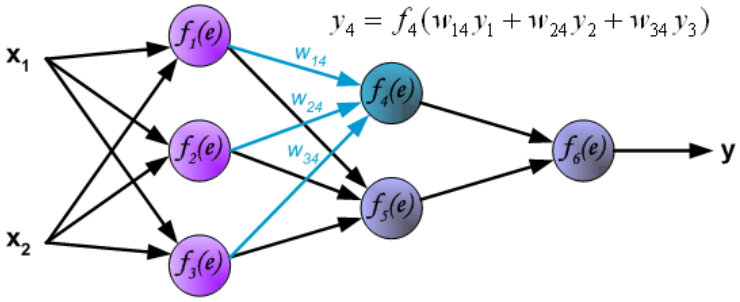

Рис.2. Пример расчета функции активации нейронов 2-го слоя

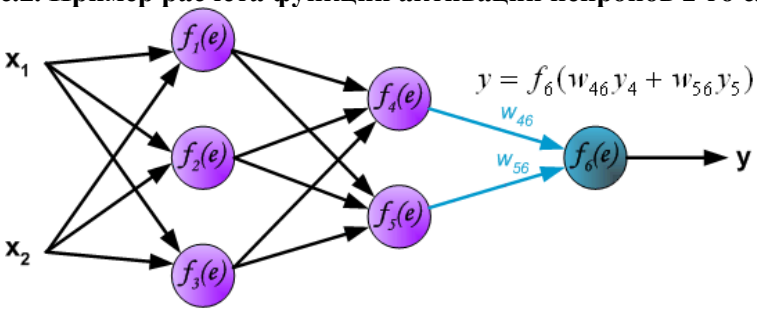

Рис. 3. Пример расчета функции активации нейронов выходного слоя

Стоит отметить, что функции активации для каждого слоя могут быть как одинаковыми, так и разными. Результатом работы нейронной сети является значение функции активации, которая в большинстве случаев имеет результат от 0 до 1. Соответственно если значении функции меньше 0,5, то нейронная сеть решила, что ответ отрицательный, больше 0,5 ответ положительный [5].

После того как сеть получила результат для ее обучения, нужно вычислить вектор ошибки. Ошибкой является разница между эталонным значением и полученным нами.

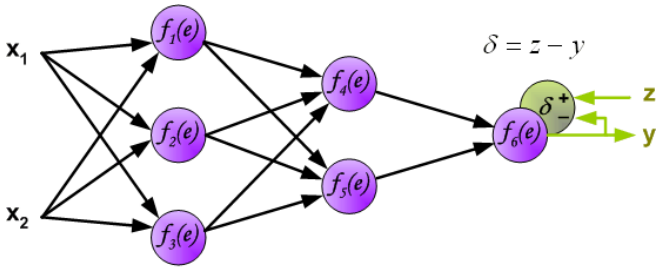

Рис. 4. Получение ошибки

ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

Далее ошибки для скрытых слоев вычисляются по следующему примеру.

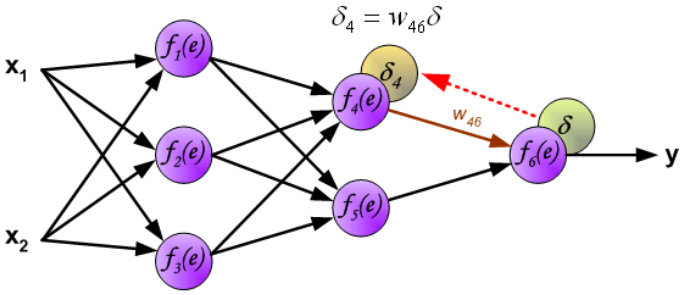

Рис. 5. Получение ошибки для 2-го слоя

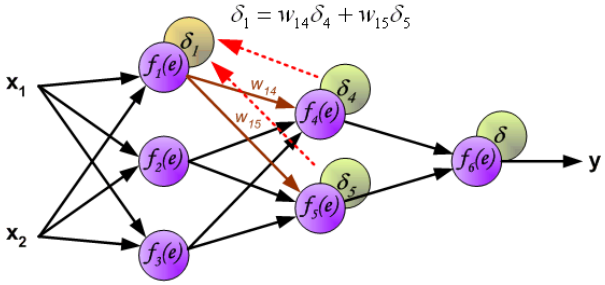

Рис. 6. Получение ошибки для 1-го слоя

Новые веса входных сигналов назначаются следующим образом:

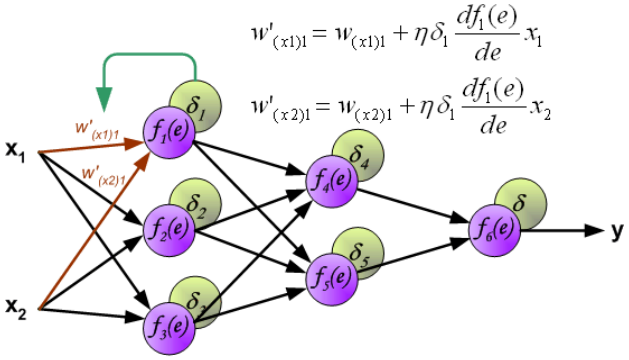

Рис. 7. Вычисление новых значений весов

Для промежуточных слоев выполняем изменение весов следующим образом.

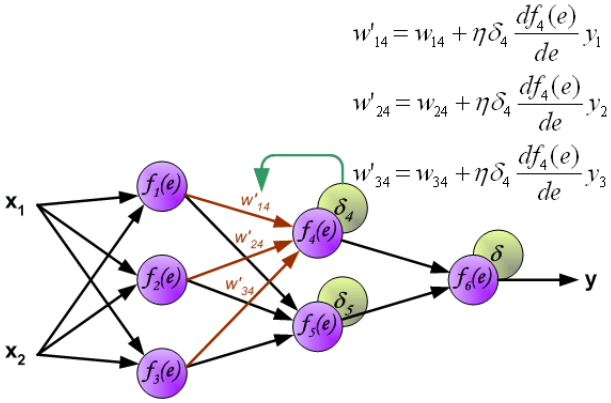

Рис. 8. Вычисление значений весов для промежуточных слоев

ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

Символом η в данных формулах обозначается скорость обучения нашей нейронной сети. Если поставить η большим, то Нейронная сеть быстро научиться распознавать лёгкие вопросы по типу тепло/холодно, но маленькие детали распознавать не сможет [6].

Рассмотрим пример создания нейронной сети на языке java. Класс, отвечающий за нейрон.

```
class Neuron
{
   protected float inputWeigth[]; // веса нейронов
   protected float outputSignal; // выходной сигнал нейрона
   protected float error; // ошибка нейрона для обучения
   Neuron(int size) // конструктор с параметром size- количество входов
   {
      inputWeigth = new float[size];
      for(int i=0;i<inputWeigth.length;i++)
      {
         inputWeigth[i] = (float) ((-0.2)+ (float) new Random().nextInt(40)/100.0);
      }
   }
}
```

Класс, отвечающий за слой нейронной сети.

```
class Layer
{
   protected ArrayList<Neuron> neurons = new ArrayList<Neuron>(); // массив нейронов
}
```

Класс, отвечающий за всю нейронную сеть.

```
public class NeuronNetwork {
protected ArrayList<Layer> layers = new ArrayList<Layer>();
   private final float learnRate = (float) 0.05; // коэффициент обучаемости
   private float FunctionActivation(float summ) {return (float) (1.0/(1.0+Math.exp(-
summ)));} // функция активации
   public  float FunctionActivationDx(double Summ) // производная от функц-//ии активации
   {
      return (float) (Summ*(1-Summ));
   }
   public void LearnNetwork(ArrayList<int[]> learnData,ArrayList<Integer> results)// Обучение нейронной сети, где learndata- массив сигналов подаваемых на вход нейрона, а results – массив требуемых выходов
   for (int i = 0; i < learnData.size(); i++) {
   for (int j = 0; j < results.get(i).length;j++) {
      for (; ; ) {
         result(learnData.get(i));
         LearnIteration(learnData.get(i), results.get(i)[j]);
         int layerCount = layers.size() - 1;
         if (Math.abs(layers.get(layerCount).neurons.get(0).error) < 0.1)   {
```

```
            break;
          }
        }
      }
    }
    }
    NeuronNetwork(int size,int neuronscount)// конструктор сети, где size- количество входов у нейронов на первом слое, а neuronscount – количество нейронов на первом слове
{
    Layer layer1 = new Layer();
    for(int i=0;i<neuronscount;i++)
    {
      layer1.neurons.add( new Neuron(size));
    }
    layers.add(layer1);
}
    public void result(int[] input)
    {
      for(int i=0;i<layers.size();i++)
      {
        for(int j = 0; j< layers.get(i).neurons.size(); j++)
        {
          if(i==0)
          {
            float summ=0;
            for(int k=0;k<layers.get(i).neurons.get(j).inputWeigth.length;k++)
            summ += layers.get(i).neurons.get(j).inputWeigth[k]*input[k];
            layers.get(i).neurons.get(j).outputSignal =
public void addLayer(int countNeurons) // добавление нового слоя
{
    Layer layer1 = new Layer();
    int size =  layers.get(layers.size()-1).neurons.size();
    for(int i=0;i<countNeurons;i++)
    {
      layer1.neurons.add( new Neuron(size));
    }
    layers.add(layer1);
}
FunctionActivation(summ);
          }
          else
          {
            float summ=0;
            for(int k=0;k<layers.get(i).neurons.get(j).inputWeigth.length;k++)
              summ += layers.get(i).neurons.get(j).inputWeigth[k]*layers.get(i-1).neurons.get(k).outputSignal;
```

```
                layers.get(i).neurons.get(j).outputSignal = FunctionActivation(summ);
            }
        }
    }
  }
  public void LearnIteration(int[] input,int result)// Одна итерация обучения нейрона
  {
    int layerCount = layers.size()-1;
    layers.get(layerCount).neurons.get(0).error = result - layers.get(layerCount).neu-
rons.get(0).outputSignal;
    for(int i=layers.size()-2;i>=0;i--)
    {
      for(int j = 0; j< layers.get(i).neurons.size(); j++)
      {
        float error=0;
        for(int k=0;k<layers.get(i+1).neurons.size();k++)
        {
          error +=  layers.get(i+1).neurons.get(k).error * layers.get(i+1).neu-
rons.get(k).inputWeigth[j];
        }
        layers.get(i).neurons.get(j).error=error;
      }
    }
    for(int i=0;i<layers.size();i++)
    {
      for(int j = 0; j< layers.get(i).neurons.size(); j++)
      {
        if(i==0)
        {
          for(int k=0;k<layers.get(i).neurons.get(j).inputWeigth.length;k++)
            layers.get(i).neurons.get(j).inputWeigth[k] += layers.get(i).neu-
rons.get(j).error * FunctionActivationDx(layers.get(i).neurons.get(j).outputSignal)*in-
put[k]*learnRate;
        }
        else
        {
          for(int k=0;k<layers.get(i).neurons.get(j).inputWeigth.length;k++)
            layers.get(i).neurons.get(j).inputWeigth[k] += layers.get(i).neu-
rons.get(j).error * FunctionActivationDx(layers.get(i).neurons.get(j).outputSignal)*lay-
ers.get(i-1).neurons.get(k).outputSignal*learnRate;
        }
      }
    }
  }
}
```

ISSN 2658 – 7505
Выпуск №5, 2019
Электронный научный журнал «Вестник молодёжной науки России»

Данную нейронную сеть можно использовать в дальнейшем для своих задач. Код написан таким образом, что есть возможность строить свою топологию нейронной сети и в дальнейшем обучать ее.

Пример обучения нейронной сети для решения задачи с использованием, написанной выше нейронной сети. Задачей будет вывод, принимать ли человека на работу в компанию, или нет. Данная нейронная сеть является примером, поэтому на входах использует 5 входных параметров:

1. Знает ли соискатель html.
2. Знает ли соискатель css.
3. Знает ли соискатель javascript.
4. Знает ли соискатель jquery.
5. Умеет ли работать с git.

Обучающая выборка, следующая, в нее заложена логика, что если соискатель не знает 2 из 3-х ключевых навыков (html, css, javascript), то он не подходит:

1. html – 1,css -1, javascript -1, jquery -1, git -1. Результат работы сети 1.
2. html – 0,css -0, javascript -0, jquery -0, git -0. Результат работы сети 0.
3. html – 0,css -0, javascript -1, jquery -1, git -1. Результат работы сети 0.
4. html – 1,css -0, javascript -1, jquery -1, git -1. Результат работы сети 1.
5. html – 1,css -1, javascript -1, jquery -0, git -0. Результат работы сети 1.

Нейронная сеть будет иметь следующую топологию:

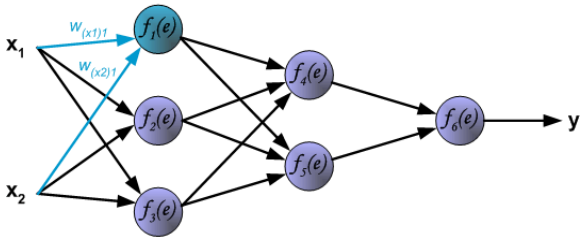

Рис. 9. Топология нейронной сети

С помощью написанного выше класса была создана НС и результаты ее обучения получились следующими.

Результаты работы нейронной сети после обучения:

1. html – 1, CSS - 1, JavaScript - 1, jQuery - 1, git - 1. Результат нужен 1, получился 0.9949747
2. html – 0, CSS - 0, JavaScript - 0, jQuery - 0, git – 0. Результат нужен 0, получился 0.0044639213
3. html – 0, CSS - 0, JavaScript - 1, jQuery - 1, git - 1. Результат нужен 0, получился 0.004430723
4. html – 1, CSS - 0, JavaScript - 1, jQuery - 1, git - 1. Результат нужен 1, получился0.9949963
5. html – 1, CSS - 1, JavaScript - 1, jQuery - 0, git - 0. Результат нужен 1, получился0.9949965.

6. На обучающей выборке нейронная сеть выдает требуемые нам результаты, однако есть еще результаты при работе с неизвестными ранее входными данными.

7. html – 0, CSS - 0, JavaScript - 1, jQuery - 1, git - 1. Результат нужен 0, получился0.0415375.

8. html – 0, CSS - 1, JavaScript - 0, jQuery - 1, git - 1. Результат нужен 0, получился0.0395445.

9. html – 1, CSS - 0, JavaScript - 0, jQuery - 1, git - 1. Результат нужен 0, получился0.9444389.

Все результаты, кроме последнего являются приемлемыми. Исходя из неверного результата можно предположить, что НС заложила в своё решение неверный вывод о том, что, если соискатель обладает знаниями html, остальные знания не важны, что не соответствует нужной нам логике. Даже на примере простой НС видно, насколько важно продумать правильно топологию НС и в особенности обучающую выборку для того, чтобы НС вела себя так, как нужно разработчику.

Заключение. Подведя итог вышесказанному можно сделать вывод о том, что нейронные сети являются перспективной и новой технологией, однако в настоящее время, имеют свои особенности и проблемы, которые в процессе разработки нужно учитывать.

Список цитируемой литературы

1. Иду по приборам: Яндекс.Такси испытало беспилотный автомобиль [Электронный ресурс] - Режим доступа: https://yandex.ru/blog/company/idu-po-priboram-yandeks-taksi-ispytalo-bespilotnyy-avtomobil, свободный. (Дата обращения: 01.12.2018 г.)
2. ИИ приходит на смену специалистам по найму персонала [Электронный ресурс] - Режим доступа: https://hightech.fm/2018/01/11/interview-with-a-robot/, свободный. (Дата обращения: 01.12.2018 г.)
3. Искусственный интеллект заменит специалистов по найму персонала [Электронный ресурс] - Режим доступа: https://robo-sapiens.ru/novosti/iskusstvennyiy-intellekt-zamenit-spetsialistov-po-naymu-personala/, свободный. (Дата обращения: 01.12.2018 г.)
4. Нейронные сети: основы теории / А. И. Галушкин. — М.: Горячая линия-Телеком, 2010. — 496 с.
5. Galushkin, Alexander I. Neural network theory, Springer, 2007. — 402 pp.
6. Сверточная нейронная сеть, часть 2: обучение алгоритмом обратного распространения ошибки [Электронный ресурс] - Режим доступа: https://habr.com/post/348028, свободный. (Дата обращения: 01.12.2018 г.)