Journal of Siberian Federal University. Engineering & Technologies 2022 15(3): 318–324
~ ~ ~



DOI: 10.17516/1999-494X-0394

УДК 629.124: 681.883

# Classification of Signals Using Neural Network Technology

**Eugenii A. Storozhok,**
**Grigorii V. Dorofeev* and Pavel A. Starodubtsev**
*Pacific Higher Naval School named after S. O. Makarov*
*Vladivostok, Russian Federation*

Received 02.02.2022, received in revised form 03.04.2022, accepted 24.04.2022

**Abstract.** Currently, the threat of sabotage on hydraulic structures (bridges, dams, drilling rigs, etc.) has increased. In this regard, the creation of effective hydroacoustic monitoring systems is an urgent task. The article considers the possibility of solving the problem of signal classification using neural network technology.

**Keywords**: modeling, neuron, neural network, learning, deterministic signal, harmonics.

Citation: Storozhok, E.A., Dorofeev, G.V., Starodubtsev, P. A. Classification of signals using neural network technology. J. Sib. Fed. Univ. Eng. & Technol., 2022, 15(3), 318–324. DOI: 10.17516/1999-494X-0394.

# Классификация сигналов с использованием технологии нейронных сетей

**Е. А. Сторожок, Г. В. Дорофеев, П. А. Стародубцев**
*Тихоокеанское высшее военно-морское училище им. С. О. Макарова*
*Российская Федерация, Владивосток*

**Аннотация.** В настоящее время возросла угроза диверсий на гидротехнических сооружениях (мосты, плотины, буровые вышки и т.п.). В связи с этим создание эффективных систем гидроакустического мониторинга является актуальной задачей. В статье рассмотрена возможность решения задачи классификации сигналов с использованием технологии нейронных сетей.

**Ключевые слова:** нейрон, нейронная сеть, обучение, детерминированный сигнал, гармоники.

© Siberian Federal University. All rights reserved
This work is licensed under a Creative Commons Attribution-Non Commercial 4.0 International License (CC BY-NC 4.0).

* Corresponding author E-mail address: grigory_artem_007@mail.ru

Цитирование: Сторожок, Е. А. Классификация сигналов с использованием технологии нейронных сетей / Е. А. Сторожок, Г. В. Дорофеев, П. А. Стародубцев // Журн. Сиб. федер. ун-та. Техника и технологии, 2022, 15(3). С. 318–324. DOI: 10.17516/1999-494X-0394.

## Введение

Реализация нейронной сети для решения какой-либо задачи включает пять основных этапов:

1. Подготовка данных для тренировки сети.
2. Создание сети.
3. Обучение сети.
4. Тестирование сети.
5. Использование сети для решения поставленной задачи [1].

В таблице 1 заданы детерминированные сигналы. Создадим нейронную сеть, которая при поступлении на входы сети гармоник одного из этих сигналов на выходе формировала бы номер сигнала, соответствующий этой гармонике. Для создания сети будем использовать систему Matlab.

Таблица 1. Детерминированные сигналы

Table 1. Deterministic signals

| № п/п | Детерминированные сигналы |
|---|---|
| 1 | sin(2*pi*5*t) |
| 2 | cos(2*pi*3*t) |
| 3 | sin(2*pi*2*t)* sin(2*pi*20*t) |
| 4 | sin(2*pi*4*t)* sin(2*pi*40*t) |
| 5 | cos(2*pi*6*t)* cos(2*pi*30*t) |
| 6 | cos(2*pi*7*t)* cos(2*pi*28*t) |
| 7 | sin(2*pi*5*t)* sin(2*pi*15*t)* sin(2*pi*30*t) |
| 8 | sin(2*pi*4*t)* cos(2*pi*24*t)* sin(2*pi*40*t) |
| 9 | cos(2*pi*3*t)*sin(2*pi*21*t)* sin(2*pi*63*t) |
| 10 | cos(2*pi*6*t)* cos(2*pi*30*t)* sin(2*pi*54*t) |

## Подготовка данных для тренировки сети

Средствами Matlab создадим массив размерностью NxM, где N=15 – количество различных гармоник из табл. 1, M – количество векторов входного массива. Размерность выходного массива будет, соответственно, 1xM. Количество входных и выходных векторов выберем равным M=100. Этого достаточно для обучения, а процесс обучения не займет много времени. Ниже приведен фрагмент программы, формирующий массивы входных и выходных векторов.

t=0:1/4000:1/2;**%** Формируем вектор времени с шагом 1/4000 с, верхний% предел 1/2 соответствует периоду гармоники с минимальной частотой 2 Гц

% Формируем векторы отсчетов сигналов

```
Y 0=sin(2*pi*5*t);
Y 1=cos(2*pi*3*t);
Y 2=sin(2*pi*2*t).* sin(2*pi*20*t);
Y 3=sin(2*pi*4*t).* sin(2*pi*40*t);
Y 4=cos(2*pi*6*t).* cos(2*pi*30*t);
Y 5=cos(2*pi*7*t).* cos(2*pi*28*t);
Y 6=sin(2*pi*5*t).* sin(2*pi*15*t).* sin(2*pi*30*t);
Y 7=sin(2*pi*4*t).* cos(2*pi*24*t).* sin(2*pi*40*t);
Y 8=cos(2*pi*3*t).*sin(2*pi*21*t).* sin(2*pi*63*t);
Y 9= cos(2*pi*6*t).* cos(2*pi*30*t).* sin(2*pi*54*t);
T=[1:10];
T=[T T T T T T T T T T];% Массив выходных векторов (номера сигналов)
n=0.2;% Шум с амплитудой 0.2
Y=Y 0;
noise=0.5*rand(1, length(t))-n;% Создаем шум с амплитудой 0.2
P=mas(Y, t, noise);
P0=P;
Y=Y 1;
noise=0.5*rand(1, length(t))-n;% Создаем шум с амплитудой 0.2
P=mas(Y, t, noise);
P1=P;
……
PP1=[P0; P1; P2; P3; P4; P5; P6; P7; P8; P9];
……
PP10=[P0; P1; P2; P3; P4; P5; P6; P7; P8; P9];
PP=[PP1; PP2; PP3; PP4; PP5; PP6; PP7; PP8; PP9; PP10];
PP=PP';% Массив входных векторов (гармоники сигналов)
%Функция mas (Y, t, noise):
function [P] = Untitled (Y, t, noise)
Y=Y+noise;% К сигналу аддитивно добавляем шум
y=fft(Y,4000);% Находим БПФ сигнала
Y=y.*conj(y)/4000;% Находим модули БПФ
P=[Y(2) Y(3) Y(4) Y(5) Y(6) Y(7) Y(15) Y(20) Y(21) Y(24) Y(28) Y(30) Y(40) Y(54) Y(63)]
End
```

**Создание сети**

Сеть строим с использованием функции newff.

```
% Построение сети
net4=newff(minmax(PP),[15,8,1],{‘purelin’ ‘logsig’ ‘purelin’},’trainlm’);
```

Параметрами функции newff являются:

PP – массив входных векторов;

15 – количество входов (нейронного входного слоя), соответствующее количеству различных гармоник из табл. 1;

8 – количество нейронов скрытого слоя (половина суммы входных и выходных нейронов);

1 – количество выходов;

'purelin' 'logsig' 'purelin' – функции активации нейронов входного, скрытого и выходного слоев соответственно;

'trainlm' – функция обучения.

## Обучение сети

Следующий шаг – обучение созданной сети. Перед обучением необходимо задать параметры обучения. Задаем функцию оценки функционирования sse.

net.performFcn='sse';

В этом случае в качестве оценки вычисляется сумма квадратичных отклонений выходов сети от эталонов. Задаем критерий окончания обучения – значение отклонения, при котором обучение будет считаться законченным:

net.trainParam.goal=0.01;

Задаем максимальное количество циклов обучения. После того как будет выполнено это количество циклов, обучение будет завершено:

net.trainParam.epochs=1000;

Теперь можно начинать обучение (рис. 1):

[net, tr]=train(net, P, T);

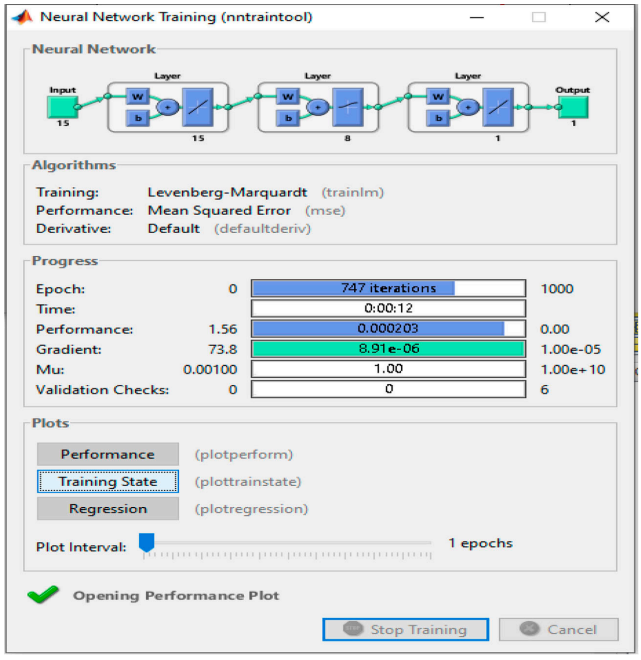


Рис. 1. Обучение сети

Fig. 1. Network training

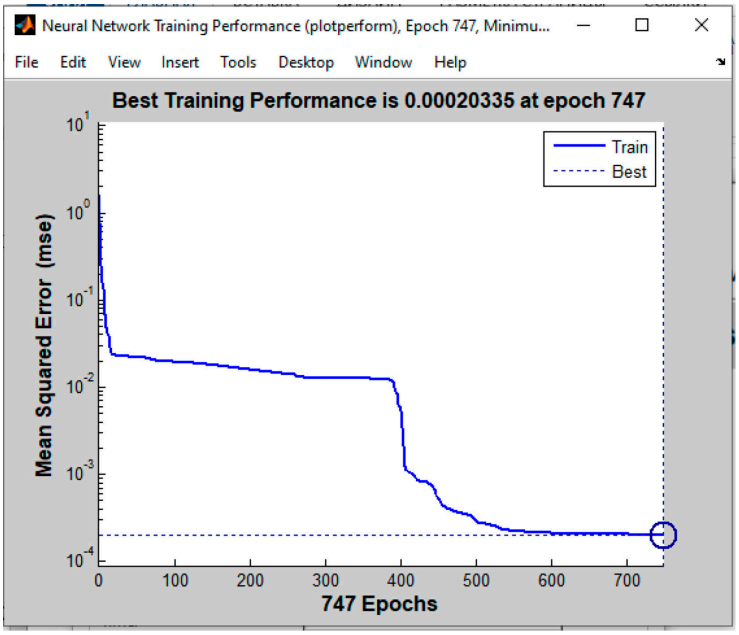


Рис. 2. Зависимость оценки функционирования от номера цикла обучения

Fig. 2. Dependence of the assessment of functioning on the number of the training cycle

Процесс обучения иллюстрируется графиком зависимости оценки функционирования от номера цикла обучения (рис. 2).

Таким образом, обучение сети окончено. Теперь эту сеть можно сохранить в файле nn1.mat:

```
save nn1 net;
```

## Тестирование сети

Перед тем как воспользоваться нейронной сетью, необходимо исследовать степень достоверности результатов вычислений сети на тестовом массиве входных векторов. В качестве тестового массива необходимо использовать массив, компоненты которого отличаются от компонентов массива, использованного для обучения. Для оценки достоверности результатов работы сети можно воспользоваться результатами регрессионного анализа, полученными при сравнении эталонных значений со значениями, полученными на выходе сети, когда на вход поданы входные векторы тестового массива. В среде MATLAB для этого можно воспользоваться функцией postreg. Следующий набор команд иллюстрирует описанную процедуру:

```
y=sim(net, P);%обработка тестового массива
[m]=postreg(y(1,:), T(1,:));%регрессионный анализ результатов обработки.
```

На рис. 3 видно, что все точки легли на прямую, что говорит о правильной работе сети на тестовом массиве.

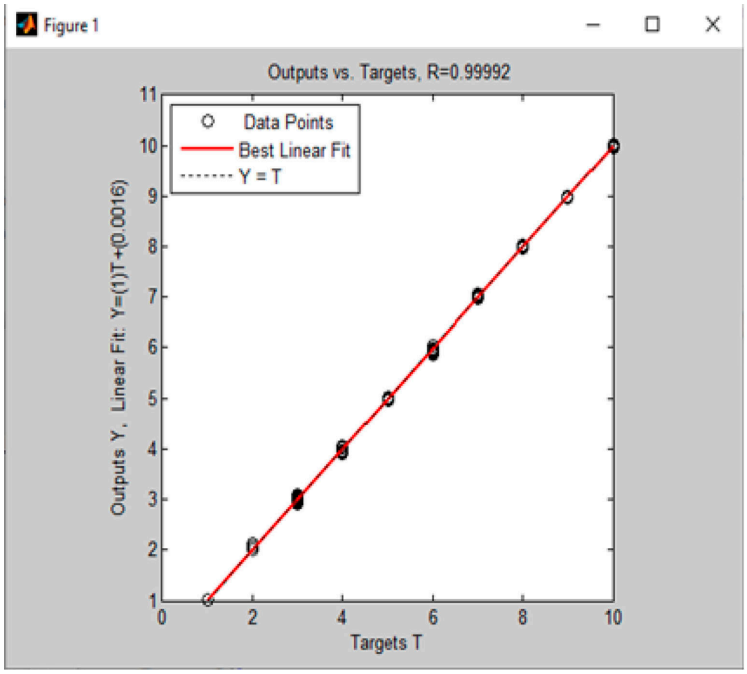


Рис. 3. Результаты тестирования

Fig. 3. Test results

**Использование сети**

Для того чтобы применить обученную сеть для обработки данных, необходимо воспользоваться функцией sim:

```
%Использование сети
m=0;%Количество ошибок
for i=1:100
Y=round(sim(net4, PP(:, i)))
if Y ~= T(i)
m=m+1;
end
end;
```

На рис. 4 показан график зависимости вероятности ошибки от амплитуды шума. Вероятность ошибки увеличивается, если амплитуда шума отличается от той амплитуды, при которой происходило обучение сети (0,2) более чем на 0,01. Следовательно, необходимо переобучение сети. На рис. 4 проиллюстрировано переобучение сети при амплитуде шума 0,29. Вероятность ошибки вновь становится близкой к нулю.


Рис. 4. Зависимость вероятности ошибки от амплитуды шума

Fig. 4. Dependence of the error probability on the noise amplitude

## Выводы

1. Нейросетевая технология позволяет эффективно и с наименьшими затратами времени осуществлять классификацию сигналов, а также выявить скрытые закономерности влияния входных параметров на выходные, что практически невозможно определить статистическими методами [2].

2. Вероятность ошибки классификации очень зависит от амплитуды шума, что вызывает необходимость переобучения нейросети.

## Список литературы / References

[1] Грудинин В.С. *Работа с нейросетями в пакете MatLab*, Киров: Издательство Вятского гос. ун-та, 2014. [Grudinin V.S. *Working with neural networks in the MatLab* package, Kirov: Publishing House of the Vyatka State University, 2014 (in Russian)]

[2] Стародубцев П.А., Сторожок Е.А., Алифанов Р.Н. Снижение вероятности ложной тревоги в измерительном узле системы гидроакустического мониторинга, *Журнал СФУ. Техника и технологии, 2020*, 13(5), 568–577. [Starodubtsev P.A., Storozhok E.A., Alifanov R.N. Reducing the probability of a false alarm in the measuring node of the hydroacoustic monitoring system, SIBFU *Journal. Engineering and technology, 2020*, 13(5), 568–577 (in Russian)]