УДК 004.8:519.6

АППРОКСИМАЦИЯ ФУНКЦИИ СИНУСА НЕЙРОННОЙ СЕТЬЮ НА ЯЗЫКЕ PHP

Климов Ю.В.

Белорусский национальный технический университет,
г. Минск, Беларусь, xpark@mail.ru

В настоящее время широкое применение получили методы моделирования технических систем на основе искусственных нейронных сетей [1][2]. Искусственная нейронной сеть представляет собой систему соединенных и взаимодействующих между собой искусственных нейронов [3]. Искусственный нейрон – это модель биологического нейрона, которая выступает в качестве элементарного процессора для простейшей обработки информации. Установлено, что нейронные сети являются нелинейными по своей природе. Во множестве задач, где линейная аппроксимация неудовлетворительна, линейные модели работают плохо.

Поэтому поставлена практическая задача аппроксимации математической функции синуса от 0 до 360 градусов (0–2 пи) с использованием нейронной сети на языке PHP без использования дополнительных библиотек [4].

Особенность нейронных сетей состоит в том, что зависимость между входом и выходом находится в процессе обучения сети. В работе использовался алгоритм управляемого обучения нейронной сети на основе обучающей выборки данных («обучение с учителем»). Обучение сети представляет собой процесс необходимой подстройки ее коэффициентов (весов), чтобы поступление входных сигналов приводило к требуемым выходным результатам.

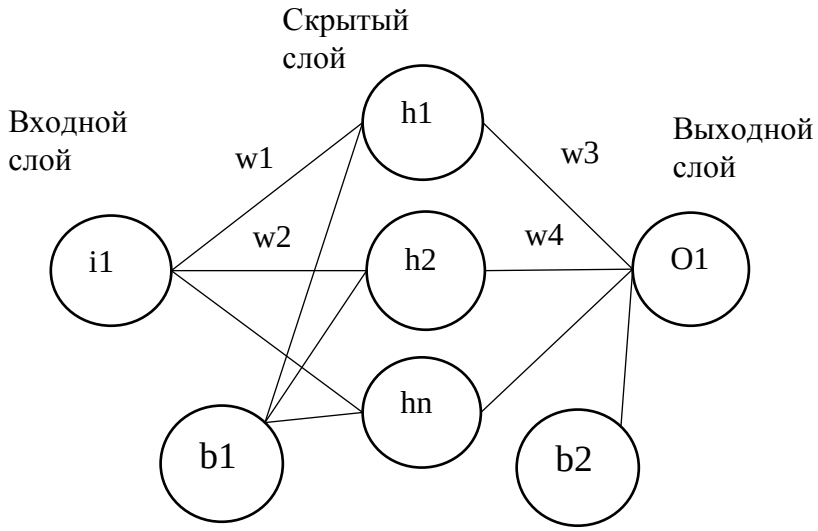


Рисунок 1 – Нейронная сеть, состоящая из 3 слоев

Для правильного решения поставленной задачи важным этапом является определение ошибки выходного сигнала сети. Ошибка (функция потерь) отражает расхождение между ожидаемым и полученным ответами. В процессе обучения нейронной сети ошибка постепенно должна уменьшаться. Ошибку можно вычислить несколькими основными способами: Mean Squared Error (MSE), Root MSE и Arctan. Чаще всего используют MSE, которая позволяет сохранить баланс между разницей и ошибкой при вычислениях.

Функция потерь по способу MSE определяется по формуле:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Ytrue - Ypred)^2 \ \#(1)$$

где n – число рассматриваемых объектов, которое в данном случае равно количеству точек на числовой прямой;

Ytrue – истинное значение переменной, то есть правильный ответ;

Ypred – предполагаемое значение переменной. Это выходной результат работы сети.

В процессе тренировки используется алгоритм оптимизации под названием стохастический градиентный спуск (SGD), который говорит, каким образом изменить вес и смещение каждого нейрона для минимизации выходных потерь нейронной сети.

Тогда можно описать потерю как многовариантную функцию:

$$L(w1, w2, w3, w4, \dots, wn, b1, b2, \dots, bn) \ \#(2)$$

Например, при изменении одного веса сети *w1* функция потерь L определяется при помощи частной производной:

$$\frac{dL}{dw1} = \frac{dL}{dYpred} \cdot \frac{dYpred}{dw1} \ \#(3)$$

Потеря L, входящая в формулу 3, определяется на основании среднеквадратичной ошибки MSE по следующей формуле:

$$L = \ (Ytrue - Ypred)^2 \ \#(4)$$

Следовательно, рассчитать частную производную $\frac{dL}{dYpred}$ возможно благодаря вышеприведенной формуле 4 потерь L:

$$\frac{dL}{dYpred} = \frac{d\ (Ytrue - Ypred)^2}{dYpred} = -2(Ytrue - Ypred) \ \#(5)$$

Выходной результат предполагаемого значения, учитывая скрытый слой *h1–hn* и нейрон смещения *b2*, определяется следующим образом:

$$Ypred = O1 = f(w3 \cdot h1 \ + w4 \cdot h2 \ + \ b2) \ \#(6)$$

где f – используемая функция активации.

$$\frac{dYpred}{dw1} = \frac{dYpred}{dh1} \cdot \frac{dh1}{dw1} \#(7)$$

Используя дифференцирование сложной функции получим:

$$\frac{dYpred}{dh1} = w3 \cdot f\,'(w3 \cdot h1 \ + w4 \cdot h2 \ + \ b2) \#(8)$$

где $f\,'$ – производная используемой функции активации.

Выполним подобные математические преобразования, также используя дифференцирование сложной функции для $\frac{dh1}{dw1}$:

$$h1 = f(w1 \cdot i1 + b1) \#(9)$$

$$\frac{dh1}{dw1} = i1 \cdot f'(w1 \cdot i1 + b1) \#(10)$$

Таким образом, частная производная $\frac{dL}{dw1}$ разбивается на ряд составных частей, которые определяются из вышеприведенных формул:

$$\frac{dL}{dw1} = \frac{dL}{dYpred} \cdot \frac{dYpred}{dh1} \cdot \frac{dh1}{dw1} \#(11)$$

Данная система расчета частных производных называется методом обратного распространения ошибки (backprop).

Рассмотрим используемые функции активации [5]. Логистическая функция (сигмоида) часто применяется в нейронных сетях в качестве функции активации [6], что позволяет усиливать слабые сигналы и уменьшать влияние сильных сигналов:

$$f(x) = \frac{1}{1 + e^{-x}} \#(12)$$

Производная сигмоиды легко выражается через саму функцию, что позволяет существенно сократить вычислительную сложность метода обратного распространения ошибки на практике:

$$f'(x) = f(x) \cdot (1 - f(x)) \#(13)$$

Важной особенностью этой функции является то, что при изменении входного значения область значений находится в пределах от 0 до 1. Это ограничивает ее использование, если необходимо выйти за пределы этих значений. Тогда для вычислений используются другие функции активации.

Также в работе использовалась еще одна активационная функция – гиперболический тангенс:

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \#(14)$$

Природа этой функции нелинейна, а диапазон значений лежит в пределах от -1 до 1. Следует отметить, что градиент тангенциальной функции больше, чем у сигмоиды (производная круче). Гиперболический тангенс является очень популярной и используемой активационной функцией при комбинации слоёв сети [7]. Производная гиперболического тангенса определяется следующим образом:

$$f'(x) = 1 - f(x)^2 \#(15)$$

Тогда изменение для веса *w1* на основании рассмотренных частных производных:

$$w1^* = w1 - learning\_rate \cdot \frac{dL}{dw1} \#(16)$$

где $learning\_rate -$ скорость обучения.

$$w1^* = w1 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot w3 \cdot f'(O1) \cdot i1 \cdot f'(h1) \right) \#(17)$$

$$w2^* = w2 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot w4 \cdot f'(O1) \cdot i1 \cdot f'(h2) \right) \#(18)$$

$$\begin{pmatrix} w1^* \\ w2^* \end{pmatrix} = \begin{pmatrix} w1 \\ w2 \end{pmatrix} - \begin{pmatrix} \frac{dL}{dYpred} \cdot w3 \cdot f'(O1) \cdot i1 \cdot f'(h1) \\ \frac{dL}{dYpred} \cdot w4 \cdot f'(O1) \cdot i1 \cdot f'(h2) \end{pmatrix} \#(19)$$

Тогда изменение для веса *w3* на основании рассмотренных частных производных:

$$w3^* = w3 - learning\_rate \cdot \frac{dL}{dw3} \#(20)$$

$$w3^* = w3 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot h1 \cdot f'(O1) \right) \#(21)$$

$$w4^* = w4 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot h2 \cdot f'(O1) \right) \#(22)$$

В данном случае это позволяет использовать матричную операцию вычитания:

$$\begin{pmatrix} w3^* \\ w4^* \end{pmatrix} = \begin{pmatrix} w3 \\ w4 \end{pmatrix} - \begin{pmatrix} \frac{dL}{dYpred} \cdot h1 \cdot f'(O1) \\ \frac{dL}{dYpred} \cdot h2 \cdot f'(O1) \end{pmatrix} \#(23)$$

Для нейрона смещения *b2* изменение веса:

$$b2^* = b2 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot \frac{dYpred}{db2} \right) \#(24)$$

$$b2^* = b2 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot f'(O1) \right) \#(25)$$

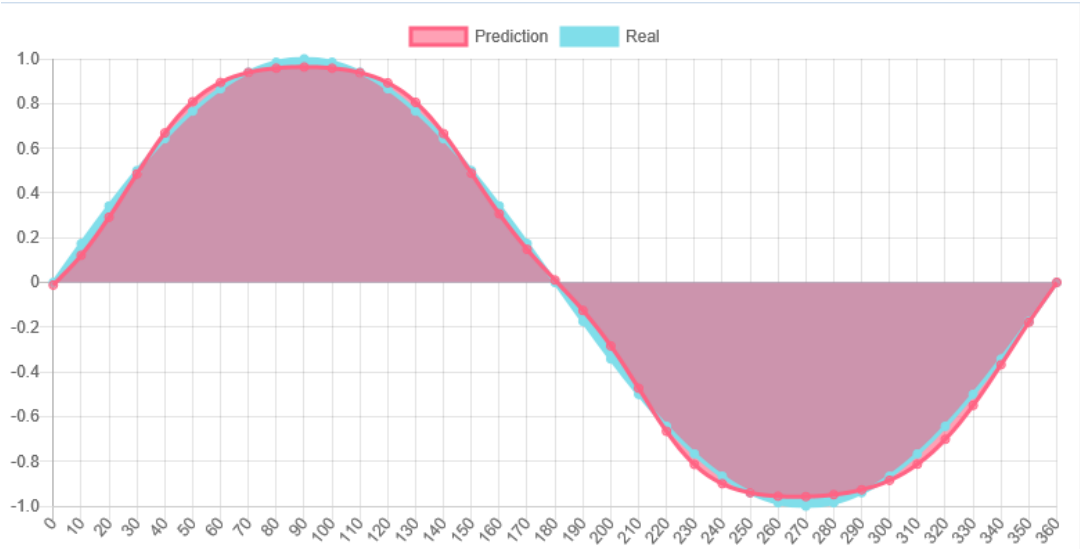
Для каждого нейрона смещения *b1* изменение веса:

$$b1^* = b1 - learning\_rate \cdot \left(\frac{dL}{dYpred} \cdot w3 \cdot f'(O1) \cdot f'(h1) \right) \#(26)$$

Разработанный алгоритм функционирует следующим образом:

1. В массив записываются значения функции *sin(x)* с шагом 5 градусов, то есть происходит формирование обучающей выборки данных;

2. Инициализируются начальные веса *w1*, *w2* и смещения *b1*, *b2* нейронной сети случайным образом по заданному закону распределения и выводятся на печать для дальнейшего сопоставления с расчетными;



-W-INI----------
Array ([0] => -0.9450175165874 [1] => 0.45191449087668 [2] => 0.097461688843305 [3] => -0.0080322125032695)
Array ([0] => 0.2445629254191 [1] => -0.14803867095524 [2] => -0.77753094107729 [3] => -0.38341150124716)
Array ([0] => Array ([0] => 0.63075844367536) [1] => Array ([0] => 0.97522643021086) [2] => Array ([0] => -0.70684985150902) [3] => Array ([0] => 0.4134098800893))
Array ([0] => -0.53451012612065)
epoch=0 mse=0.42040487500351
epoch=100 mse=0.27501529802851
epoch=200 mse=0.12071649074135

Рисунок 2 – Результаты аппроксимации функции синуса

3. Создается объект на языке PHP и вызывается функция forward(), в которую передаются выбранные и сформированные начальные параметры. В этой функции задается скорость обучения, выполняется заданное количество эпох и итераций во вложенных циклах по сформированной ранее выборке данных. Дополнительно создан класс Matrix для ряда матричных операций, который активно использовался в php-скрипте. Класс нейронной сети включает методы расчета функций и производных сигмоиды, а также гиперболического тангенса. Класс наследует созданный класс Matrix, поэтому все его методы доступны для использования;

4. По достижению количества эпох значению, кратному заданному (например, 100), вызывается функция прямого распространения feed_forward() в которую передаются измененные веса и смещения с выходным расчетом массива сети. Далее производится расчет ошибки MSE по вышеприведенной формуле. Ошибка выводится php-скриптом через заданное количество эпох, что позволяет контролировать процесс обучения нейронной сети;

5. На основе метода обратного распространения ошибки циклично осуществляется изменение весов и смещений сети, которые выводятся в массив;

6. Предсказание (аппроксимация) значений функции *sin(x)* осуществляется при помощи функции feed_forward() на основании уже рассчитанных весов и смещений сети, а также исходных значений в градусах с заданным шагом.

Для построения графиков на основании массивов полученных данных в php-скрипте дополнительно использована JavaScript-библиотека, позволяющая выбрать цветовые оттенки и шаг построения нескольких графиков для наглядности представления результатов.

Таким образом, решена задача аппроксимации математической функции синуса с использованием многовариантных вычислений при помощи нейронной сети. Алгоритм можно использовать в реальном масштабе времени, разместив php-скрипт на хостинге в сети Интернет. Время обучения нейронной сети определяется количеством принятых эпох и находится в допустимых пределах ограничений для работы скриптов на web-сервере.

Список литературы:

1. Климов, Ю.В. Прогнозирование экономических показателей в бизнесе организации / Ю.В. Климов // Электронная библиотека БГУ [Электронный ресурс]. – Режим доступа: http://elib.bsu.by. – Дата доступа: 10.11.2020.
2. Климов, Ю.В. Моделирование и прогнозирование экономических показателей работы транспортной системы / Ю.В. Климов // Белорусский государственный университет транспорта (БелГУТ) [Электронный ресурс]. – Режим доступа: https://www.bsut.by/science/conferences/novosti/novosti-3/transport-febt/transport-febt-matherials. – Дата доступа: 10.11.2020.
3. Нейронная сеть [Электронный ресурс] // Википедия – свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Нейронная_сеть. – Дата доступа: 10.11.2020.
4. PHP [Электронный ресурс] // Википедия – свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/PHP. – Дата доступа: 10.11.2020.
5. Функция активации [Электронный ресурс] // Википедия – свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Функция_активации. – Дата доступа: 10.11.2020.
6. Сигмоида [Электронный ресурс] // Википедия – свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Сигмоида. – Дата доступа: 10.11.2020.
7. Гиперболические функции [Электронный ресурс] // Википедия – свободная энциклопедия. – Режим доступа: https://ru.wikipedia.org/wiki/Гиперболические_функции. – Дата доступа: 10.11.2020.