Ссылка на статью:
// Ученые записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2020, № 2, с. 80-85.

Поступила: 28.11.2020
Окончательный вариант: 05.12.2020

УДК 004.032.26

Применение машинного обучения для экспресс-тестирования уровня разработчиков

Щенникова Е.В.*[*], *Навошин Р.Е.*, *Трифонова Д.Ю.

[*]schennikova8000@yandex.ru
МГУ им. Н.П. Огарева, Саранск, Россия

Рассмотрена реализация метода машинного обучения с использованием нейронной сети с прямой связью на основе использования библиотеки Brain.js. Предложено построение искусственной нейронной сети, которая автоматизирует процесс определения уровня опыта соискателя на должность в IT-компании с учетом небольших затрат времени и ресурсов. Результаты могут найти применение при разработке программного обеспечения для экспресс-тестирования.

***Ключевые слова**: машинное обучение, JavaScript, Brain.js, искусственная нейронная сеть, экспресс-тестирование.*

Введение

К числу актуальных задач, которые приходится решать специалистам отдела кадров IT-компаний, относится задача найма сотрудников. Для того чтобы провести рекрутинговую кампанию, привлекаются несколько специалистов компании. Сначала HR-специалисты занимаются поиском кандидатов и проводят первичное собеседование. Далее подключаются технические специалисты для проведения технического интервью. Все эти процессы занимают много времени. Проблема состоит в том, что у технических специалистов отнимается время от их непосредственных задач по разработке ПО или создается дополнительная нагрузка, при этом успешность найма не гарантируется. Поэтому компания несет определенные риски в данной сфере. Основная задача состоит в том, чтобы с учетом затрат небольшого количества времени оценить уровень опыта кандидата. С помощью современных информационных технологий и машинного обучения можно автоматизировать этот процесс и оптимизировать траты ресурсов в этой сфере.

Решение задач и построение систем машинного обучения на сегодняшний день является одним из самых популярных, актуальных и современных направлений, которое строится на стыке информационных технологий, математического анализа и статистики [1]. Всё шире охватывая сферы деятельности современного человека, который применяет основанные на методах искусственного интеллекта пользовательские продукты, машинное обучение развивается и увеличивает сферы своего применения.

В проблемах кадровой политики можно выявить определенные закономерности, сформировать статистику и для решения подобного рода проблем применить машинное обучение. На первом этапе настоящей работы составлен список важных тем, которые должны знать кандидаты на определенную IT-должность. На втором этапе был подготовлен список популярных вопросов для каждой темы на основании технических интервью различных компаний. На третьем этапе осуществлена классификация полученного списка вопросов по категориям сложности. Следующим шагом стало формирование обучающей и тестовой выборки для будущей системы путем опроса реальных разработчиков. После получения и анализа необходимой информации была построена модель с обучением на полученных данных. В статье на основе применения метода машинного обучения NeuralNetwork из библиотеки Brain.js предложено построение искусственной нейронной сети, которая автоматизирует процесс определения уровня опыта кандидата на должность за минимальное количество времени.

1. Инструменты для решения задач машинного обучения

Для решения поставленной задачи была выбрана нейронная сеть с прямой связью (Feedforward Neural Network). Эта сеть хорошо подходит для решения задач классификации некоторых объектов, но она не имеет памяти о предыдущих действиях и имеет бесконечное разнообразие результатов. Но для поставленной задачи эти ограничения не критичны, так как не влияют на конечный результат [2].

В качестве языка программирования был выбран язык JavaScript. Он достаточно простой и подходит для большого количества задач, в том числе и для машинного обучения [3].

В качестве среды выполнения JavaScript была выбрана Node.js – кросс-платформенная среда с открытым исходным кодом для разработки серверных и сетевых приложений [4]. Среда Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API (реализованный на C++), подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода. В Node.js используется Stream – концепция, с помощью которой обрабатываются данные небольшими частями, что позволяет задействовать небольшой объем оперативной памяти. Это ускоряет обработку больших данных.

Нейронная сеть с прямой связью реализована как метод в библиотеке Brain.js. Данная библиотека является достаточно быстрой, так как она выполняет вычисления с использованием графического процессора [2]. Важно отметить, что в ходе исследования была за-

действована возможность визуализации нейронной сети в виде создания SVG-изображения. Обученная нейронная сеть была сохранена в формате JSON для того, чтобы не обучать ее заново при каждом новом использовании. Это полезная функциональность библиотеки Brain.js, так как процесс обучения нейронной сети может быть довольно затратным по производительности.

2. Обучающая выборка

Для создания обучающей выборки был сформирован список из 20 вопросов. Далее были опрошены 50 действующих разработчиков компании разного уровня опыта. Итоговая выборка была разделена следующим образом: 35 сотрудников для обучения и 15 сотрудников для тестирования (соответственно 70% и 30 %). Уровень опыта разработчика был разбит на 3 стадии: 1 – junior (начинающий), 2 – middle (средний) и 3 – senior (опытный).

На рис. 1 в качестве примера представлена часть обучающей выборки.

	A	B	C	D	E	F	G	H	I
1	senior	0	0	1	1	1	1	1	1
2	junior	0	0	1	0	0	1	0	0
3	middle	0	1	0	1	0	1	0	1
4	middle	0	0	1	1	1	0	0	1
5	middle	1	0	0	1	0	1	0	1
6	junior	0	0	1	1	0	0	0	0
7	senior	1	0	1	1	1	1	1	1
8	senior	1	1	1	1	0	1	1	1
9	junior	0	1	0	0	1	0	0	1
10	middle	1	1	0	0	0	0	1	1

Рис. 1. Пример обучающей выборки.

В первой колонке указывается предполагаемый уровень разработчика, который проходил опрос. В строке указывается список ответов на вопросы, где 1 – это правильный ответ, 0 – неправильный.

Для реализации нейронной сети с прямой связью был использован метод NeuralNetwork из библиотеки Brain.js [2]. При использовании данного метода библиотеки каждый обучающий шаблон должен иметь вход и выход. На вход подается массив ответов на вопросы, а на выход – уровень сотрудника.

Для тестирования реализуемой сети на вход подается массив ответов сотрудника. Далее сеть возвращает результат в виде объекта, в котором перечислены все возможные варианты уровней сотрудника и число от 0 до 1. Чтобы определить уровень сотрудника, выбираем самое большое значение. Пример использования метода в коде представлен на рис. 2.

```
const brain = require('brain.js');

const config = {
  activation: 'sigmoid', // supported activation types: ['sigmoid', 'relu', 'leaky-relu', 'tanh'],
};

const net = new brain.NeuralNetwork(config);

net.train( data: [
  { input: [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1], output: { senior: 1 } },
  { input: [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0], output: { senior: 1 } },
  { input: [1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1], output: { middle: 1 } },
  { input: [1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0], output: { middle: 1 } },
  { input: [1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0], output: { junior: 1 } },
  { input: [0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0], output: { junior: 1 } },
]);

const output = net.run( data: [1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]);

// Пример вывода результата тестового запуска
// {
//   senior: 0.8579725623130798,
//   middle: 0.11179767549037933,
//   junior: 0.05669515207409859
// }
```

Рис. 2. Пример использования метода NeuralNetwork.

Согласно рис. 2 данный случай можно интерпретировать следующим образом: протестированный сотрудник с вероятностью, равной 0,85, имеет уровень «senior». Далее рассмотрим структуру нейронной сети, использованной для решения данной задачи.

3. Искусственная нейронная сеть

Структура разработанной нейронной сети представлена на рис. 3. Зеленым цветом обозначены нейроны входного слоя, количество нейронов соответствует длине входного массива (количество вопросов). В данном слое обработка данных не производится, только распределение по нейронам следующего уровня. Красным цветом обозначены нейроны скрытого слоя, количество подбирается автоматически. Синему цвету соответствуют нейроны выходного слоя, количество равно количеству вариантов выхода (количеству уровней сотрудника).

В конфигурации метода NeuralNetwork можно указать функцию активации. В ходе реализации было произведено сравнение доступных в библиотеке Brain.js функций активации: сигмоиды (sigmoid), выпрямителя (relu), гиперболического тангенса (tanh) [5]. В каждом отдельном случае тестового запуска сети точнее могла быть любая функции активации из перечисленных. Но в среднем, на всей тестовой выборке точнее была сигмоида. Выпрямитель мог вообще не дать результата в некоторых отдельных элементах тестовой выборки. Это связано с тем, что обучающая выборка довольно небольшая. Но сигмоида и

гиперболический тангенс отработали на всей тестовой выборке. Там, где выпрямитель не выдал результата, остальные две функции хоть и с маленькой вероятностью, но выдали правильный ответ.

В реализованной нейронной сети (рис. 3) функция активации нейрона определяет выходной сигнал, который определяется входным сигналом или набором входных сигналов. Стандартная компьютерная микросхема может рассматриваться как цифровая сеть функций активации, которые могут принимать значения «ON» (1) или «OFF» (0) в зависимости от входа. Это похоже на поведение линейного перцептрона в нейронных сетях. Однако только нелинейные функции активации позволяют таким сетям решать нетривиальные задачи с использованием малого числа узлов. В искусственных нейронных сетях эта функция также называется передаточной функцией. Такой функцией в предлагаемом решении стала сигмоидная функция активации, т.к. она показала самую высокую точность на различных тестовых данных.

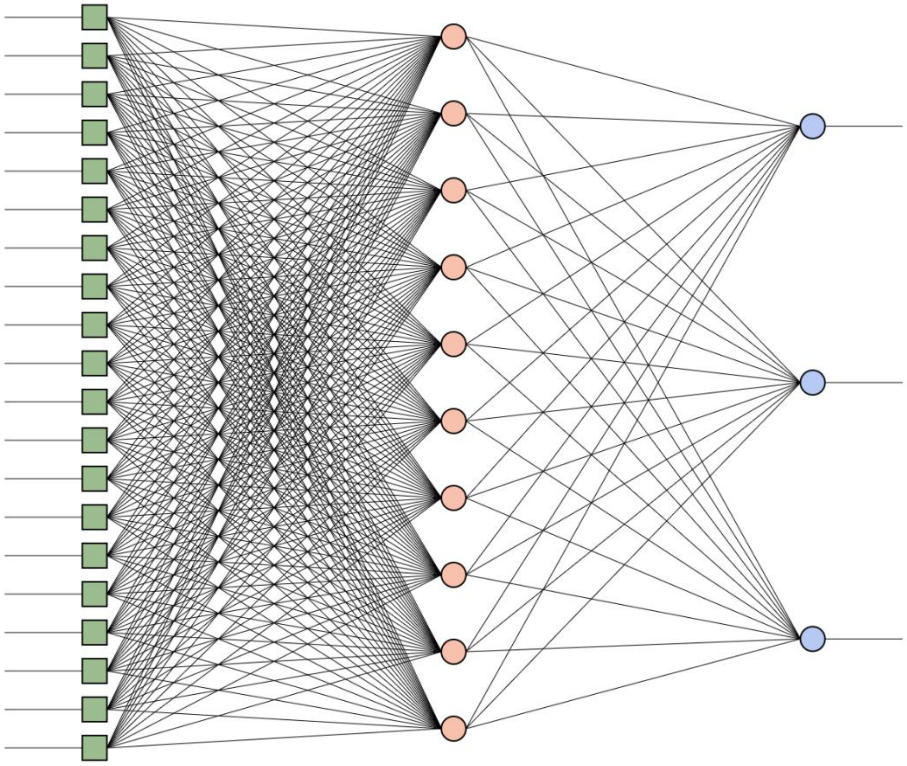

Рис. 3. Структура нейронной сети, полученная в ходе разработки.

Для обучения нейронной сети были указаны следующие параметры конфигурации:

- допустимый процент ошибок – 0.5%;
- количество итераций – 20000.

Разработанная нейронная сеть прекращает обучение, когда-либо процент ошибок стал менее 0.5%, либо прошло 20000 итераций.

Заключение

По результатам работы нейронной сети на тестовой выборке точность определения уровня сотрудника оказалась 90 – 95%. Так как нейронная сеть с прямой связью имеет бесконечное количество вариантов результата, то показатель точности может меняться при каждом запуске сети, но это не критично для данной задачи. Это достаточно хороший показатель для такой небольшой обучающей выборки. Была проведена серия экспериментов с различными параметрами конфигурации сети. В результате был предложен экспериментальный вариант программного обеспечения для экспресс-тестирования при найме кандидатов в IT-компанию.

Список литературы

1. Попков Ю.С. Машинное обучение и рандомизированное машинное обучение: сходство и различие // *Системный анализ и информационные технологии (Труды Восьмой международной конференции САИТ-2019, Иркутск, 08–14 июля 2019).* Москва: Федеральный исследовательский центр «Информатика и управление» Российской академии наук, 2019. С.10–15.
2. *Brain.js.* Режим доступа: https://github.com/BrainJS/brain.js (дата обращения 01.03.2020).
3. *JavaScript*. Режим доступа: https://ru.wikipedia.org/wiki/JavaScript (дата обращения: 05.04.2020).
4. *Node.js.* Режим доступа: https://ru.wikipedia.org/wiki/Node.js (дата обращения 05.04.2020).
5. *Функция активации.* Режим доступа: http://www.wikiwand.com/ru/Функция_активации (дата обращения: 15.09.2020).