УДК 004.853

БИБЛИОТЕКА МАШИННОГО ОБУЧЕНИЯ TENSORFLOW

А.А. СОЛОВЬЁВ
(Представлено: канд. физ.-мат. наук, доц. Ю.Ф. ПАСТУХОВ)

Рассматривается библиотека с открытым исходным кодом TensorFlow, предназначенная для решения задач построения и тренировки искусственных нейронных сетей, на примере классической задачи распознавания рукописных цифр.

TensorFlow — открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия [1, 2].

В качестве примера, на основе которого будут продемонстрированы некоторые возможности библиотеки, будет решаться классическая задача классификации рукописных цифр. В статье будут описаны основные шаги при взаимодействии с библиотекой, а также показаны результаты построенной и обученной нейронной сети.

В данной статье используется набор данных MNIST [3], который содержит 70 000 изображений рукописных цифр с низким разрешением (28 на 28 пикселей) в оттенках серого в 10 категориях (рис. 1). Этот набор данных относительно мал и используется для проверки того, что алгоритм работает так, как ожидалось. Он является хорошей отправной точкой для тестирования и отладки кода.

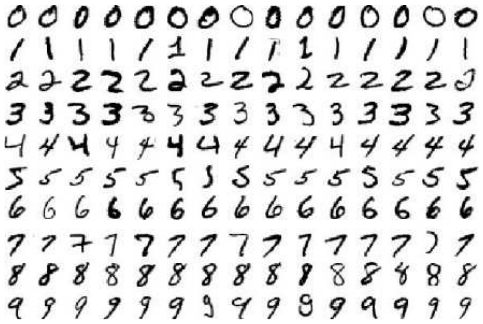

Рисунок 1. – База данных MNIST

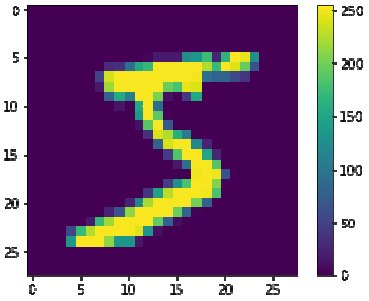

Рисунок 2. – График значений пикселей

Основными шагами, которые необходимо проделать для классификации изображений являются:

– Импорт набора данных MNIST.
– Предварительная обработка данных.
– Построение модели.
– Настройка слоев.
– Компиляция модели.
– Обучение модели.
– Оценка точности.
– Классификация.

В качестве примера будет использоваться 60 000 изображений для обучения сети и 10 000 изображений, для оценки того, насколько точно сеть научилась классифицировать изображения. Получить доступ к базе данных MNIST можно непосредственно из TensorFlow, просто импортировав и загрузив данные:

```
1:    (train_images, train_labels), (test_images, test_labels) =
            keras.datasets.mnist.load_data()
```

Загрузка набора данных возвращает четыре массива NumPy [4]:

– Массивы train_images и train_labels — это наборы данных, которые использует модель для обучения.
– Модель тестируется на тестовом наборе, массивах test_images и test_labels.

Изображения представляют собой массивы 28x28 NumPy, значения пикселей которых находятся в диапазоне от 0 до 255. Метки представляют собой массив целых чисел от 0 до 9, которые соответствуют классу цифры. Каждое изображение отображается на одну метку.

Перед подготовкой сети данные должны быть предварительно обработаны. Если проверить первое изображение в наборе обучающих данных, то можно увидеть, что значения пикселей находятся в диапазоне от 0 до 255 (рисунок 2). Перед подачей изображений в модель нейронной сети необходимо значения всех пикселей выборки привести к диапазону от 0 до 1. Для этого значение каждого пикселя необходимо поделить на 255. Важно, чтобы наборы данных для обучения и тестов были предварительно обработаны таким же образом.

Построение нейронной сети требует настройки слоев модели, а затем компиляции модели.

Основным строительным блоком нейронной сети является слой. Слои извлекают представления из данных, подаваемых в них. Большая часть глубокого обучения состоит в объединении простых слоев. Большинство слоев, таких как tf.keras.layers.Dense, имеют параметры, которые изучаются во время обучения.

Первый слой нашей сети tf.keras.layers.Flatten преобразует формат изображений из 2d-массива (28 на 28 пикселей) в 1d-массив из 28 * 28 = 784 пикселей. У этого слоя нет параметров для изучения; он только переформатирует данные.

После того, как пиксели сплющены, сеть состоит из последовательности трёх слоёв tf.keras.layers.Dense и одного слоя tf.keras.layers.Dropout. Слои Dense — это плотно связанные или полностью связанные нейронные слои. Первый слой Dense содержит 256 узлов (или нейронов), а второй 128 узлов. Данные слои используют функции активации ReLU. После данных слоёв следует Dropout слой, который отсеивает нейроны для предотвращения переобучения нейронной сети. Последний полносвязный слой с 10-узлами содержит функцию активации softmax, который возвращает массив из десяти вероятностных оценок, сумма которых равна 1. Каждый узел содержит оценку, которая указывает вероятность того, что текущее изображение принадлежит одному из 10 классов.

```
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(256, activation=tf.nn.relu),
    keras.layers.Dense(128, activation=tf.nn.relu),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

Прежде чем модель будет готова к обучению, ей потребуется еще несколько настроек. Они добавляются во время этапа компиляции модели:

– Оптимизатор влияет на то, как модель обновляется на основе данных, которые она видит.

– Функция потери — это мера, которая указывает насколько будет точна модель во время обучения.

– Метрики — используются для контроля за этапами обучения и тестирования.

Чтобы начать обучение нейронной сети необходимо вызвать метод model.fit. Данный метод в качестве параметров принимает массив тренировочных изображений, их метки и количество тренировочных эпох.

Для оценки качества обученной нейронной сети предусмотрена функция evaluate, которая принимает массивы тестовых изображений и их меток. Данная функция возвращает значения, показывающие точность классификации сети на тестовых изображениях.

```
model.compile(optimizer=tf.train.AdamOptimizer(),
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(train_images, train_labels, epochs=5)

test_loss, test_acc = model.evaluate(test_images, test_labels)
```

В нашем случае модель обучилась достаточно хорошо и имеет показатель точности 0,98.

Обученную модель можно использовать для прогнозирования изображений. Предсказание модели представляет собой массив из 10 чисел. Они описывают «уверенность» модели в том, что изображение

соответствует каждому из 10 разных цифр. Мы можем видеть, какая метка имеет наибольшее доверительное значение на рисунке 3.

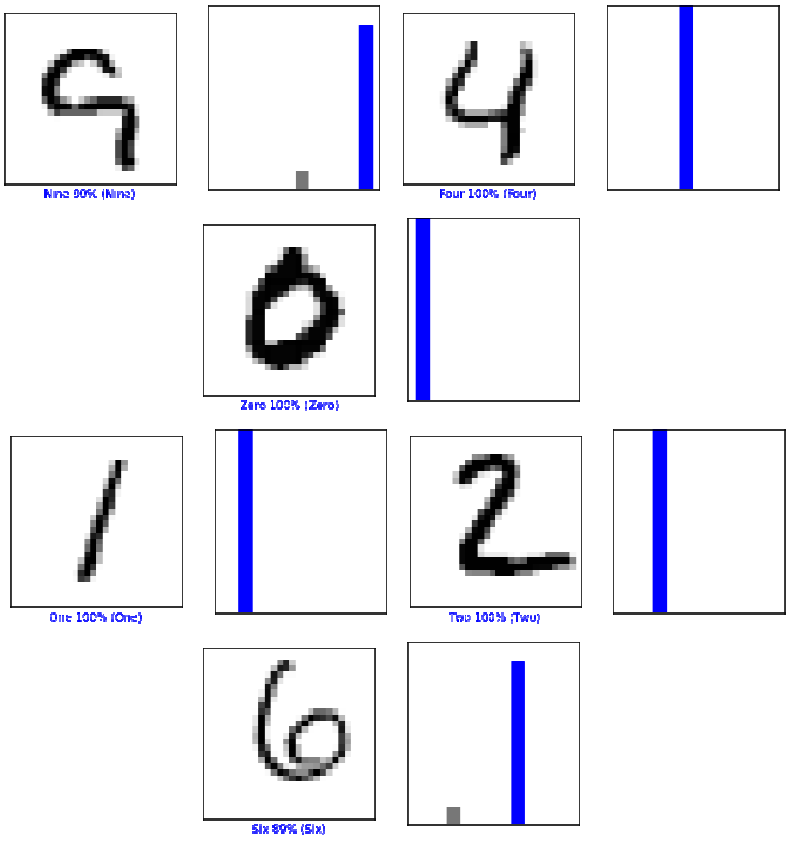

Рисунок 3. – Показатели уверенности обученной нейронной сети

На рисунке 3 можно увидеть, что модель уверенно определила рукописные цифры 4, 0, 1 и 2, и имела сомнения на примерах цифр 9 и 6. Данные сомнения сети возникли из-за не очевидного шрифта данных цифр.

В заключении статьи можно сказать, что рассмотренная библиотека TensorFlow обладает обширными методами и функциями построения нейронных сетей. Для построения и обучения моделей в TensorFlow используется tf.keras [6], API высокого уровня. В данной библиотеке реализованы функции для построения моделей любой сложности, а также содержащих различные слои с соответствующими параметрами. Вокруг библиотеки имеется огромное сообщество людей, которые делятся опытом создания нейронных сетей, а также помогают улучшать функционал самой библиотеки. TensorFlow является современным отличным инструментом в создании, обучении и использовании различных нейронных сетей.

ЛИТЕРАТУРА

1. TensorFlow [Электронный ресурс]. — Режим доступа: https://ru.wikipedia.org/wiki/TensorFlow. — Дата доступа: 24.09.2018.
2. Открытая программная библиотека TensorFlow [Электронный ресурс]. — Режим доступа: https://www.tensorflow.org. — Дата доступа: 24.09.2018.
3. База данных рукописных цифр MNIST [Электронный ресурс]. — Режим доступа: http://yann.lecun.com/exdb/mnist/. — Дата доступа: 24.09.2018.
4. Библиотека с открытым исходным кодом NumPy [Электронный ресурс]. — Режим доступа: http://www.numpy.org. — Дата доступа: 24.09.2018.
5. Открытая нейросетевая библиотека Keras [Электронный ресурс]. — Режим доступа: https://keras.io. — Дата доступа: 24.09.2018.
6. Графическая библиотека MatPlotLib [Электронный ресурс]. — Режим доступа: https://matplotlib.org. — Дата доступа: 24.09.2018.