Дипломный проект

по курсу Data Science Foundation на тему «Машинное обучение с учителем: классификация текста с помощью нейронных сетей»

Описание проекта

Основна мета та проблематика проекту

▶ Используя несистематизированную информацию, касающуюся строительных нормативов ФРГ и имеющую вид электронной документации в формате PDF, получить данные в виде готового текста (нормативных таблиц) для последующего использования на Вёб-портале при расчёте калькуляций различных строительных работ

Опис вхідних даних

 Электронные документы с нормативной строительной документацией в виде файлов PDF и сканов в различном графическом формате

Моделі ML, що застосовані

Машинное обучение с учителем на основании нейронной сети

Описание машинной модели

Инструментом для распознавания был использован Tesseract OCR (оптического распознавания символов). Плюсами данной библиотеки можно отметить обученные языковые модели (>192), разные виды распознавания (изображение как слово, блок текста, вертикальный текст), легкая настройка.
 OCR использует нейронные сети для поиска и распознавания текста на

OCR использует нейронные сети для поиска и распознавания текста на изображениях.

Tesseract ищет шаблоны в пикселях, буквах, словах и предложениях, использует двухэтапный подход, называемый адаптивным распознаванием. Требуется один проход по данным для распознавания символов, затем второй проход, чтобы заполнить любые буквы, в которых он не был уверен, буквами, которые, скорее всего, соответствуют данному слову или контексту предложения.

Общие принципы алгоритма «распознавалок текста»

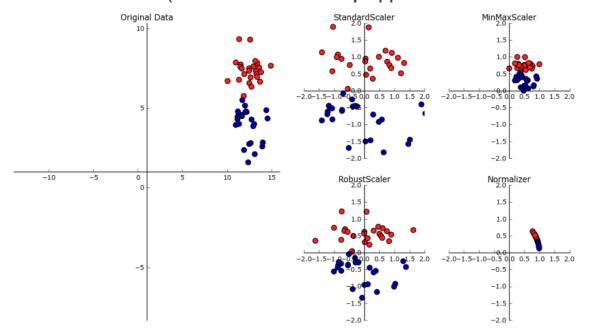
- https://habr.com/ru/post/466565/
 На основании набора графических изображений букв и символов формируем данные для обучения:
 - выделяем каждую букву в графическом формате градаций серого,
 предварительно отмасштабировав её до нужного размера (например, квадрата 28х28 пикселей)
 - сохраняем каждую букву в условном словаре в виде кортежа (tuple)(x, w, letter)
 соответствие матрицы пикселей и буквы в таблице символов
 - ▶ Буквы готовы для распознавания, распознавать их мы будем с помощью сверточной нейронной сети этот тип сетей неплохо подходит для таких задач. Исходный датасет изображений букв не делаем сами, а берём, к примеру, EMNIST(The EMNIST dataset is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset) имеет 62 разных символа (А.. Z, 0.. 9 и пр)





https://en.wikipedia.org/wiki/MNIST_database

▶ В двух словах - берётся готовая библиотека изображений символов (EMNIST), выполняется предварительная обработка данных под выбранный тип нейронной сети (нормализация или что-то другое) и формируем train и test set'ы (сами символы представляют собой обычные массивы):



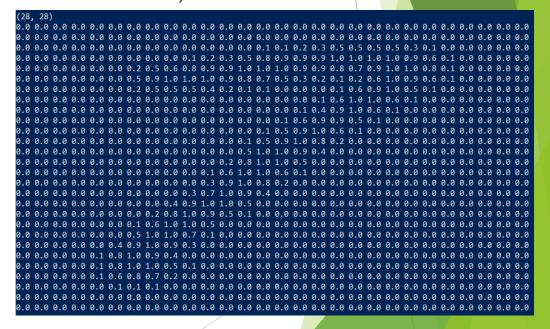


Рис. 3.1 Различные способы масштабирования и предварительной обработки данных

Общие принципы алгоритма «распознавалок» - продолжение:

 Запускаем обучение сети, в конце процесса сохраняем обученную модель на диск:

```
learning_rate_reduction =
keras.callbacks.ReduceLROnPlateau(monitor='val_accuracy', patience=3,
verbose=1, factor=0.5, min_lr=0.00001)
model.fit(X_train, x_train_cat, validation_data=(X_test, y_test_cat),
callbacks=[learning_rate_reduction], batch_size=64, epochs=30)
model.save('emnist_letters.h5')
```

Для распознавания мы загружаем модель и вызываем функцию predict_classes (этот метод мы предварительно реализовали у себя на Python).

HELLØ WØRLD

```
model = keras.models.load_model('emnist_letters.h5')
s_out = img_to_str(model, "hello_world.png")
print(s_out)

PS C:\Python> python .\keras_emnist.py
Using TensorFlow backend.
```

Небольшие выводы по работе алгоритма

- Забавная особенность нейронная сеть «перепутала» букву «О» и цифру «О», что впрочем, неудивительно т.к. исходный набор EMNIST содержит рукописные буквы и цифры, которые не совсем похожи на печатные. В идеале, для распознавания экранных текстов нужно подготовить отдельный набор на базе экранных шрифтов, и уже на нем обучать нейросеть.
- Например, в том же Tesseract можно увидеть «уверенность», с которой алгоритм определяет слова(набор символов):

```
val api = Tesseract()
```

val image =

ImageIO.read(URL("http://img.ifcdn.com/images/b313c1f095336b6d681f75888f8932 fc8a531eacd4bc436e4d4aeff7b599b600_1.jpg"))

val result = api.getWords(preparedImage, ITessAPI.TessPageIteratorLevel.RIL_WORD)

[WHEN [Confidence: 94.933418 Bounding box: 48 251 52 14], SHE [Confidence:

95.249252 Bounding box: 109 251 34 15], CATCHES [Confidence: 95.973259 Bounding

box: 151 251 80 15], YOU [Confidence: 96.446579 Bounding box: 238 251 33 15],

CHEATING [Confidence: 96.458656 Bounding box: 117 278 86 15]]



Реализация прототипа Web-приложения

- ► Создано SPA вёб-приложение на основе Python-фреймворка Flask (Django лучше, но для SPA немого быстрее для создания Flask).
- Функциональность приложения:
 - ▶ Загрузка PDF-файла
 - ▶ Конвертация его страниц в отдельный image
 - Распознавание текста для последующей обработки в клиентском вёбприложении

Интерфейс приложения:

