Финальное задание по Deep Learning Basic

Общее описание

В рамках задание предлагается самостоятельно обучить модель OCR для распознавания капчи.

Пример входа:

Пример выхода: 2b827

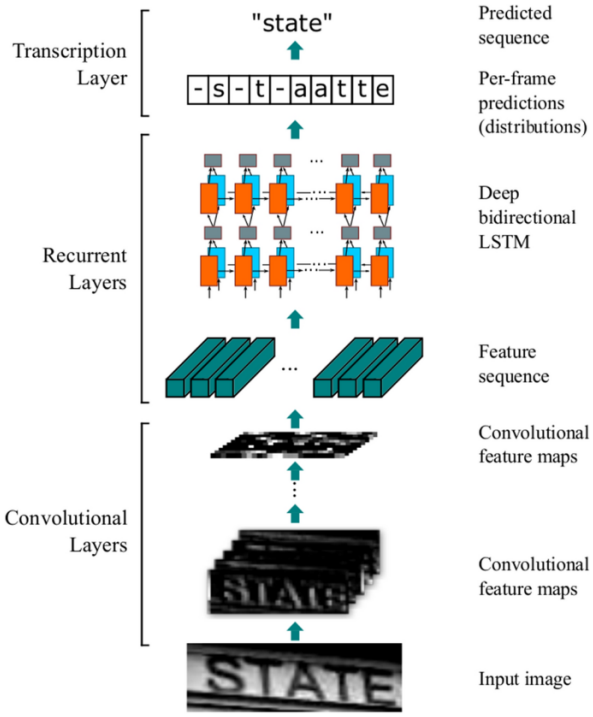
Задание сдается в виде github репозитория (с правами public), в котором содержится либо ноутбук, либо скрипты обучения/оценки с файлом описания решения и финальными метриками.

Корпус данных

Предлагается использовать корпус картинок капчи. Скачать его можно по ссылке: https://disk.yandex.ru/d/JQn56xLQ_3QPHw.

Предлагаемая архитектура решения

Предлагается реализовать модель, состоящую из двух блоков: Fully-convolutional CNN (FCNN) и Bi-LSTM. На выходе предлагается использовать кросс-энтропийный критерий. Пример архитектуры представлен ниже (выход закодирован под CTCLoss, не обращайте внимания):



Обратите внимание на формат выхода CNN и формат входа в RNN слоях. Обычно требуются дополнительные манипуляции с тензором (permute, reshape). Не обязательно реализовывать указанный выше подход. Любое решение с хорошим качеством распознавания подойдёт (residual networks, attention, трансформеры и др.). Главное ограничение – нельзя использовать предобученные веса модели.

Что должна включать работа

Для сдачи проекта нужно подготовить ноутбук с решением. В каждом решении должны быть описанные ниже блоки.

Подготовка данных. Нужно реализовать класс данных (наследник torch.utils.data.Dataset). Класс должен считывать входные изображения и выделять метки из имён файлов. Для чтения изображений предлагается использовать библиотеку Pillow. Дирректория содержит набор данных, который необходимо разделить на тренировочную и тестовую выборки в отношении четыре к одному.

Создание и обучение модели. Код модели должен быть реализован через слои стандартной библиотеки torch (torchvision.models и аналоги использовать нельзя). Поскольку число символов в капче фиксировано, можно использовать обычный кросс-энтропийный критерий. Желающие могут использовать и CTC-loss. Цикл обучения можно реализовать самостоятельно или воспользоваться библиотеками PyTorch Lightning / Catalyst.

Подсчет метрик. После обучения нужно оценить точность предсказания на тестовой выборке. В качестве метрики предлагается использовать долю неверно распознанных символов, Character Error Rate (CER).

Анализ ошибок модели. В этой секции нужно найти изображения из тестового корпуса, на которых модель ошибается сильнее всего (по loss или по CER). Предлагается выписать в ноутбук возможные причины появления этих ошибок и пути устранения.

Оценка работы

Работа будет оцениваться по следующим критериям:

1. Качество кода
2. Качество реализованного подхода
3. Качество описания принимаемых решений и анализа ошибок
4. Финальные метрики