1. Результаты.

Задача 1.

Задача представляет собой задачу детекции объектов на изображении.

Первый вариант решения.

Первое, что пришло в голову – это обнаружение чёрных рамок через поиск контуров. Но этот способ не работает на всех изображениях. Способ плохо работал там, где красная сетка и там, где совсем темное изображение (нечеткий переход от черных рамок к другим частям изображения). Чтобы найти рамки на картинках, где нанесена красная сетка пришлось добавить Гаусовское размытие. Но всё равно в итоге получилось не очень.

Поэтому был использован второй вариант.

Второй вариант решения.

Использовать предобученную нейросеть и с помощью трансфер лёнинга научить её находить черные рамки. Этот способ работал уже лучше. Дообученная сеть неплохо находила черные рамки, кроме тех изображений, где всё изображение — это черный экран. Уверен, что если докинуть такие черные картинки в набор на котором происходило обучение, то сеть будем хорошо работать и с черными экранами.

Задача 2.

Задача представляет собой задачу классификации изображений. В наличии 10 классов (и в основной задаче, и в дополнительной).

Когда увидел, что в данных MNIST – сразу стало понятно, что нужно применять свёрточные нейросети, хорошо зарекомендовавшие себя в работе с изображениями. В результате написал и обучил сверточную сеть архитектуры Lenet. Сеть научилась классифицировать изображения с цифрами. Метрика Accuracy на проверочной части получилась 0.97

Дополнительное задание тоже сделал через архитектуру Lenet. Метрика Accuracy на проверочной части получилась 1.

1. Используемые инструменты.

Для первой задачи в первом варианте использовался язык программирования python, модуль opencv, среда разработки Jupyter Notebook. Во втором варианте использовался фреймворк pytorch, нейросеть resnet50. Так как это задача детекции, то нужно было размечать данные для дообучения нейросети. Я использовал инструмент разметки **vgg image annotator**. Этот инструмент я использовал ранее на финальном хакатоне Цифрового прорыва в декабре 2021 года (перед хакатоном пришлось посмотреть несколько инструментов аннотации). Нейросеть resnet50 я использовал потому что уже применял её дообучение ранее на хакатоне Норильского никеля и в финале Цифрового прорыва. Обучение нейросети проводилось в google colab на gpu. Для обучения я разметил 85 картинок, взяв по нескольку из каждой папки (60img, 600img, 6000img) и каждой подпапки. В качестве оптимизатора был выбран SGD. Скорость обучения была фиксированной. Веса модели сохранялись в отдельную папку для возможного последующего использования. В обучении использовалось фиксированное количество эпох. После обучения я тестировал нейросеть на картинках, которые она ранее не видела.

Для второй задачи использовался python, pytorch, Jupyter Notebook. На pytorch была реализована сверточная нейросеть архитектуры Lenet. Обучение проводилось на cpu рабочего пк. Классы в доп задании имеют одинаковое количество примеров, по 10 000 примеров в каждом. В основном задании, где MNIST тоже примерно одинаковое количество примеров в каждом классе. Это идеальные условия, когда классы сбалансированы. При сбалансированных классах можно применять метрику Accuracy (при несбалансированных нет). В качестве оптимизатора был выбран Adam, он как правило, хорошо работает из коробки и его обычно рекомендуют использовать в первую очередь. Скорость обучения была фиксированной, но её можно было бы сделать динамически изменяющейся в зависимости от эпохи с помощью schedule. Я не стал делать сохранение весов, так как модель обучалась быстро, но можно было бы сохранить с помощью torch.save. В обучении использовалось фиксированное количество эпох, но можно было бы сделать early stopping, и остановить обучение, когда лосс на проверочной части прекратит снижаться определенное количество эпох.

1. Перспективные инструменты.

Для первой задачи можно было бы попробовать вариант с поиском перепадов цветов. Найти среднее значение цвета по каждой строке и столбцу в картинке. У областей с черными полями среднее значение было бы 0 (чуть больше для картинок с красной сеткой). У областей вне черных рамок средний цвет был бы гораздо больше 0. Потом найти на каких по счету пикселях в строке и столбце идет перепад с 0 до больших значений, это были бы границы. Но такой способ будет медленным из-за подсчета среднего по каждой строке столбцу в картинке (в нашем примере 704 пикселя по высоте и 1280 по ширине), но подозреваю, что более точным, чем поиск контуров.

Вместо resnet50 можно было бы попробовать какие-то другие архитектуры.

Для второй задачи можно было бы попробовать кроме Lenet какие-нибудь другие архитектуры свёрточных нейросетей (например, облегченную версию VGG).

1. Информация.

Условие первой задачи было сформулировано не очень понятно (лично для меня), не сразу понял. Упоминаемые размеры рамок отвлекают от сути. Можно было бы просто написать, что нужно найти черные поля по бокам изображений. Задания были прикольные. Рад, что удалось их сделать.

1. Ссылка на репозиторий.

Ссылка на репозиторий на гитхабе (там код и текстовое описание)

<https://github.com/The-Illusive-Man-2000/test_assignment>

Ссылка на гугл диск (там текстовое описание и веса модели)

<https://drive.google.com/drive/folders/1f209M6Dp1-9I8De1xwZYIsIT56lrv3U6?usp=sharing>