Весь раздел «Обработка данных» закомментирован. Непонятно его нужно смотреть или нет?

*# def normalize(picture):*

*# width, height = picture.size*

*# normalized\_array = []*

*# for j in range(0, height):*

*# for i in range(0, width):*

*# pixel = picture.getpixel((i,j))*

*# normalized\_array.append( pixel[0] / 255.0 )*

*# return np.array(normalized\_array)*

Очень странный код. Во-первых таким образом мы двухмерный массив превращаем в одномерный (или трехмерный в двухмерный), во-вторых почему нельзя весь массив разделить на 255? Это же не C++, а python.

*if (indexVal[0] == 0):*

*# # print(f"{file} is a Car")*

*# pass*

*# else:*

*# os.remove(dir\_from+file)*

Хороший способ очистки данных. Еще надежнее было бы применить голосование нескольких сетей для сегментации автомобилей на изображении.

x = BatchNormalization()(x)

x = Dropout(0.25)(x)

x = Dense(256, activation = 'relu')(x)

x = BatchNormalization()(x)

Сейчас обычно не добавляют никаких слоев между global max pool и выходным слоем. Но нейронные сети это в большой степени алхимия, так что ваш подход тоже может оказаться хорошим, а может наоборот оказаться хуже. Самое главное – больше слоев не значит лучше, а если это полносвязные слои, то обычно оказывается наоборот.

Кроме того, если у вас модель – это последовательность слоев, то почему не использовать Sequential вместо Model?

Валидация у вас делается без аугментаций, это хорошо, обычно допускают ошибку: делают валидацию с аугментациями, и в итоге сравнение моделей получается некорректным.

**for** \_ **in** range(EPOCHS):

predictions\_tta.append(model.predict(sub\_generator, verbose=1))

Тут можно еще делать так: сделать сначала 10 попыток на каждом изображении, а затем делать дополнительные попытки на тех изображениях, в которых результат остался спорным.

*Ансамблирование. Больший вес отдадим лучшей модели, т.к. у нее лучшая метрика.*

Спорный вопрос. Лучшая матрика на валидационном датасете не означает лучшую метрику на сабмите. Все-таки с кажлой эпохой сеть обучается, я бы отдавал веса поровну или наоборот больше той, что последняя. Но здесь однозначной правды нет, опять-таки: алхимия.

*> !!! На полный ноутбук не хватает квоты в 10 часов GPU процессора*

Можно сохранять результаты в kaggle-датасет, затем загружать из датасета и продолжать обучение.

**Комментарии по коду [SF] Car Classification with new data.ipynb**

Как я понял, в ноутбуке тот же код, просто больше данных. Хорошо что вы собрали дополнительные данные, обычно ленятся это делать :)