Здравствуйте! Далее комментарии по коду работы:

Почему-то в коутбуке нет ячеек кода, поэтому если там есть какие-то графики или подсчет точности, то не смогу посмотреть.

```
!unzip /kaggle/input/sf-dl-car-classification/train.zip -d /kaggle/temp
!unzip /kaggle/input/sf-dl-car-classification/test.zip -d /kaggle/temp
```

Здесь можно добавлять флаг -q, чтобы имена файлов не печатались в консоль (это еще и замедляет процесс разархивирования).

```
model.add(Dense(256))
model.add(BatchNormalization())
model.add(Dense(10))
```

Между линейными слоями (dense) должна быть функция активации, иначе нет имеет смысла добавлять слои, то есть Dense(256), Dense(10) будет эквивалентно просто Dense(10).

```
optimizer=Adam(ExponentialDecay(1e-3, 100, 0.9)),
```

С ExponentialDecay нужна осторожность, потому что если LR слишком быстро затухает, то сеть может не обучиться, даже если у сети был потенциал к дальнешему обучению. У вас BATCH_SIZE=14, а изображений около 15 тысяч, поэтому одна эпоха (один проход по всем изображениям) занимает около 1000 батчей, за это время LR умножится на 0.9^(1000/100)=0.35. После 5-7 эпох LR станет настолько низким, что сеть перестанет обучаться.

Как вариант, можно использовать InverseTimeDecay или ReduceLROnPlateau.

Можно было попробовать TTA, это помогло бы еще немного повысить точность.

Отзыв подготовил ментор проекта Олег Седухин. Если есть вопросы, можете задать их в канале #0-project_7-ford_vs_ferrari, постараюсь помочь разобраться. Успехов в дальнейшем обучении! Обязательно подключайтесь на итоговый созвон-вебинар по проекту **26 февраля**. Анонс вебинара появится позже в канале #0-project_7-ford_vs_ferrari.