Постановка задачи.

Необходимо реализовать собственной CallBack, и провести обучение вашей модели из практического занятия №6 с написанным CallBack'ом. То, какой CallBack необходимо реализовать определяется вариантом.

Вариант 2

Построение и сохранение карты признаков на заданных пользователем эпохах. Карта признаков - ядро свертки представленное в виде изображения. Название карты признака должно иметь вид <номер слоя>_<номер ядра в слое>_<номер эпохи>

Выполнение работы.

Создан класс *FeatureMapSaver*, унаследованный от *Callback*. В конструктор он принимает список номеров эпох. В нем переопределен метод *on_epoch_end*. В начале метода проверяется, входит ли текущая эпоха в список интересующих эпох. Производится итерация по всем слоям модели, проверяется, является ли слой типа *Convolutional2D*. Все ядра сверточных слоев сохраняются в виде изображения.

```
class FeatureMapSaver(Callback):
    def __init__(self, epochs: list):
        super().__init__()
        self.epochs = epochs
        os.makedirs('./feature_maps', exist_ok=True)

    def on_epoch_end(self, epoch, logs=None):
        if epoch not in self.epochs:
            return

        for index, layer in enumerate(self.model.layers):
            if type(layer) == Convolution2D:
                weights = layer.weights[0]

                for i in range(weights.shape[2]):
                    for j in range(weights.shape[3]):
                        image = Image.fromarray(np.uint8(weights[:, :, i,
j] * 255))
                        image.save(f'./feature_maps/{index}_{i *
weights.shape[3] + j}_{epoch}.png')
```

Затем данный Callback передается в качестве параметра в функцию fit с указанием пятой эпохи. За одну эпоху сохраняется более 400 карт признаков, поэтому к отчету прилагаются только несколько.