МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №5
по дисциплине «Искусственные нейронные сети»
Тема: Распознавание объектов на фотографиях

Студентка гр. 8382 _____ Кузина А.М.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2021

Цель работы

Распознавание объектов на фотографиях (Object Recognition in Photographs) CIFAR-10 (классификация небольших изображений по десяти классам: самолет, автомобиль, птица, кошка, олень, собака, лягушка, лошадь, корабль и грузовик).

Задачи:

- Ознакомиться со сверточными нейронными сетями
- Изучить построение модели в Keras в функциональном виде
- Изучить работу слоя разреживания (Dropout)

Требования:

1. Построить и обучить сверточную нейронную сеть
2. Исследовать работу сеть без слоя Dropout
3. Исследовать работу сети при разных размерах ядра свертки

Ход работы

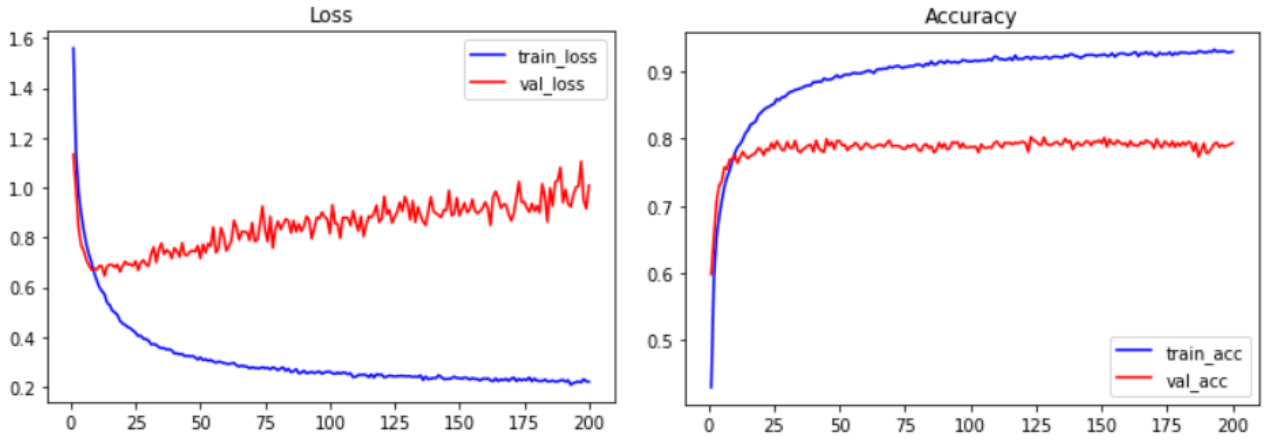
Рассмотрим изначальную модель сверточной сети, которая позволяет находить на изображении шаблоны независимо от их расположения. Она состоит из четырех слоев свертки, двух слоев субдискретизации и двух полносвязных слоев. Сверточные слои выделяют наиболее полезные признаки в изображении и для каждого такого признака формируют карту признаков. Слой субдискретизации уменьшает разрешение изображения, тем самым сокращая количество параметров модели. Полносвязные слои осуществляют непосредственно классификацию.

Параметры модели:

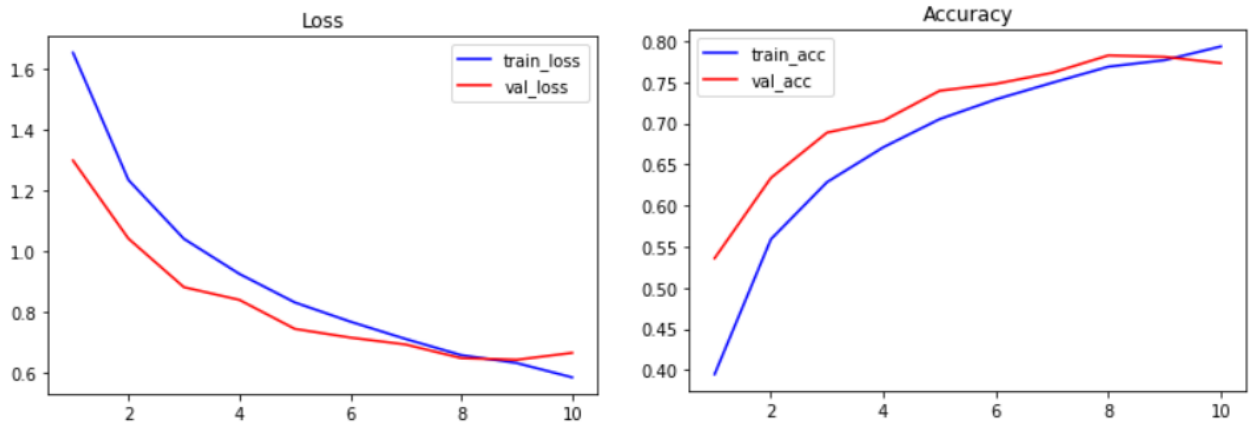
batch_size = 32 – размер батча, num_epochs = 200 – количество эпох
kernel_size = 3 – размер ядра свертки для сверточных слоев
pool_size = 2 – размер окна субдискретизации, то во сколько раз уменьшится каждое из имерений изображения
conv_depth_1 = 32 – число признаков, которые вычисляет 1й сверточный слой
conv_depth_2 = 64 – число признаков, которые вычисляет 2й сверточный слой
drop_prob_1 = 0.25 – вероятность не рассмотрения нейрона на 2х Dropout слоях
drop_prob_2 = 0.5 – вероятность не рассмотрения нейрона на 3м Dropout слое
hidden_size = 512

```
inp = Input(shape=(depth, height, width))

conv_1 = Convolution2D(conv_depth_1, kernel_size, padding='same', activat
ion='relu')(inp)
conv_2 = Convolution2D(conv_depth_1, kernel_size, padding='same', activat
ion='relu')(conv_1)
pool_1 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_2)
drop_1 = Dropout(drop_prob_1)(pool_1)

conv_3 = Convolution2D(conv_depth_2, kernel_size, padding='same', activat
ion='relu')(drop_1)
conv_4 = Convolution2D(conv_depth_2, kernel_size, padding='same', activat
ion='relu')(conv_3)
pool_2 = MaxPooling2D(pool_size=(pool_size, pool_size))(conv_4)
drop_2 = Dropout(drop_prob_1)(pool_2)

flat = Flatten()(drop_2)
hidden = Dense(hidden_size, activation='relu')(flat)
drop_3 = Dropout(drop_prob_2)(hidden)
out = Dense(num_classes, activation='softmax')(drop_3)

model = Model(inputs=inp, outputs=out)
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
```

В результате обучения сети и валидации была получена точность 60%. На графике потерь видно, что после 10 эпох сеть переобучается:

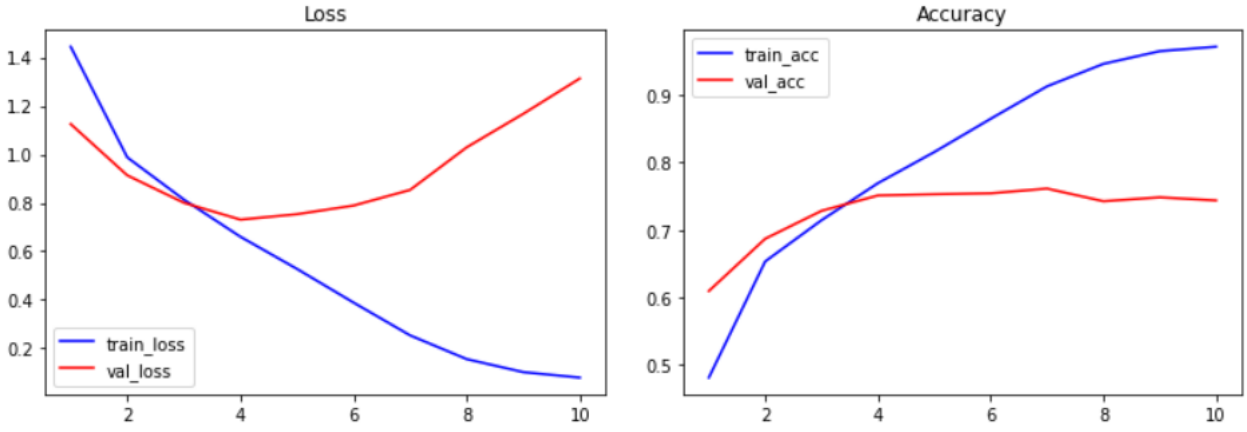


Поэтому количество эпох было сокращено до 10, при этом точность выросла до 75% и при повторных запусках не падала ниже 70%, ниже представлены графики точности и потерь сети:



Также данная сеть была протестирована с разными значениями размера батча: 32, 64, 128 и 256 образцов, рассмотренные результаты были очень похожи, одна при размере батча 256 точность начинала падать, поэтому решено было оставить размер батча 128, это значительно ускоряет время обучения. При такой конфигурации сети точность предсказаний в среднем выше 73%.

Далее все Dropout были удалены из архитектуры сети, чтобы определить их влияние на работу сети. Рассмотрим полученные результаты после обучения такой сети:

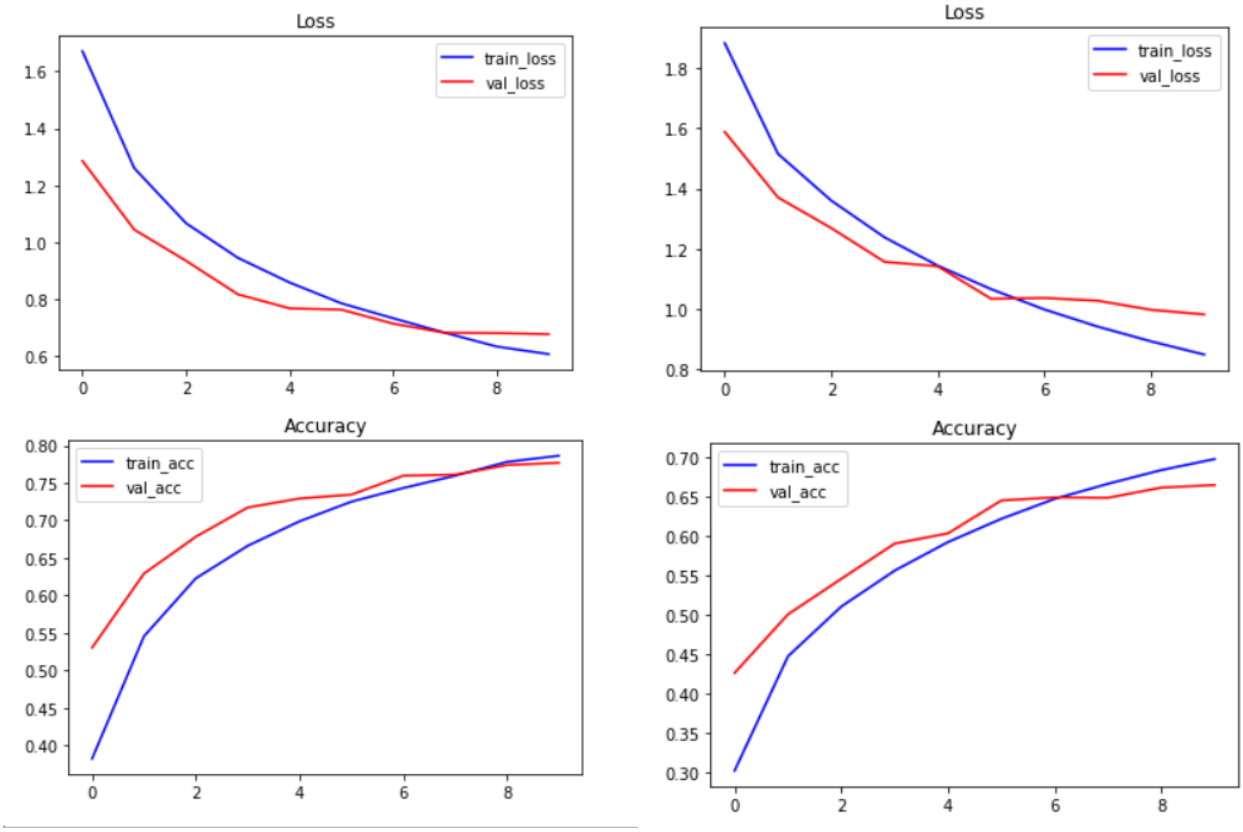


На графиках видно, что точность валидационных данных сильно падает и модель переобучается уже после 5й эпохи, в сравнении с архитектурой, в которой данные слои присутствуют. Очевидно что, наличие слоев Dropout в данной архитектуре сети важно для ее корректного обучения.

Далее архитектура сети со слоями прореживания была рассмотрена с различными размерами ядра свертки.

В сравнении изначального размера ядра 3 на 3, ядро 2 на 2 дало немного меньшую точность, вероятно потому что такое ядро слишком маленькое, чтобы замечать, выделять важные признаки. Однако при размерах ядра 5 на 5, 7 на 7 и 9 на 9 точность с каждым большим ядром падает все сильнее.

Графики потерь и точности для ядер 5 на 5(слева) и 9 на 9 (справа) представлены ниже:



Видно, как различаются точность и ошибка для этих двух ядер – чем меньше ядро, тем более точный результат. Результаты ядра 3 на 3 оказались самыми оптимальными для данной задачи.

Выводы

В данной лабораторной работе была создана нейронная сеть со сверточной архитектурой, классифицирующая изображения. Также изучен принцип работы сверточных нейронных сетей, исследовано влияние слоя Dropout на процесс обучения, рассмотрена работа нейронной сети при различных размерах ядра свертки.