Система по распознаванию автомобильных номеров

**Дизайн архитектуры**

В этой части будет описана полная архитектура решения.

Постоянная работа обеспечивается за счёт запущенного потока.

Класс *GFMain* запускает тред, который позволяет постоянно опрашивать камеру на наличие новых фотографий через http.

При наличии новой фотографии, фотография обрабатывается функцией *predict*, файла *sekkar*.

def predict(self, image):

На вход приходит изображение, на выходе мы имеем строку в формате:

return '{"status": 1, "number": "AER964", "frame": "514,215;627,219;626,241;513,236", "confidence": 100}'

Сама функция представляет собой, последовательность двух нейронных сетей. Первая сегментирует плату номера, вторая преобразует плату в текст.

В первую очередь изображение нормализуется, чтобы значения входного тензора (картинки) были одного порядка со значениями инициализированных весов.

Затем изображение передается в первую нейронную сеть. 1-я нейросеть решает задачу сегментации номера. Ее выход - маска напротив номера. Пример рисунок 1.

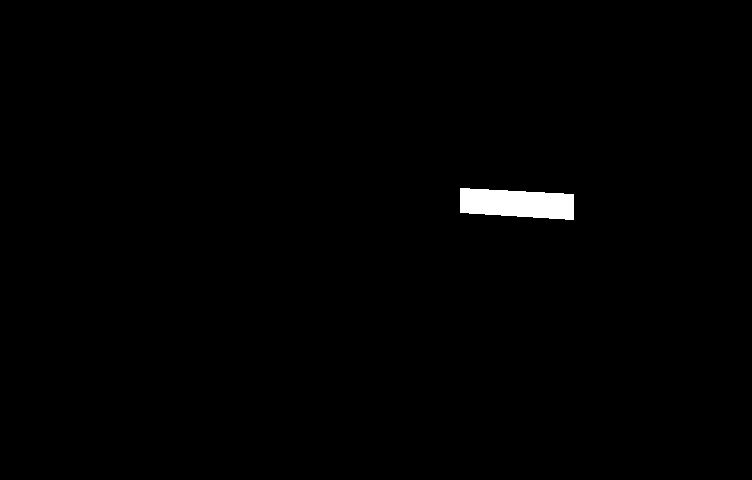


Рисунок 1. Пример сегментации

Данная нейронная сеть представляет из себя UNET.

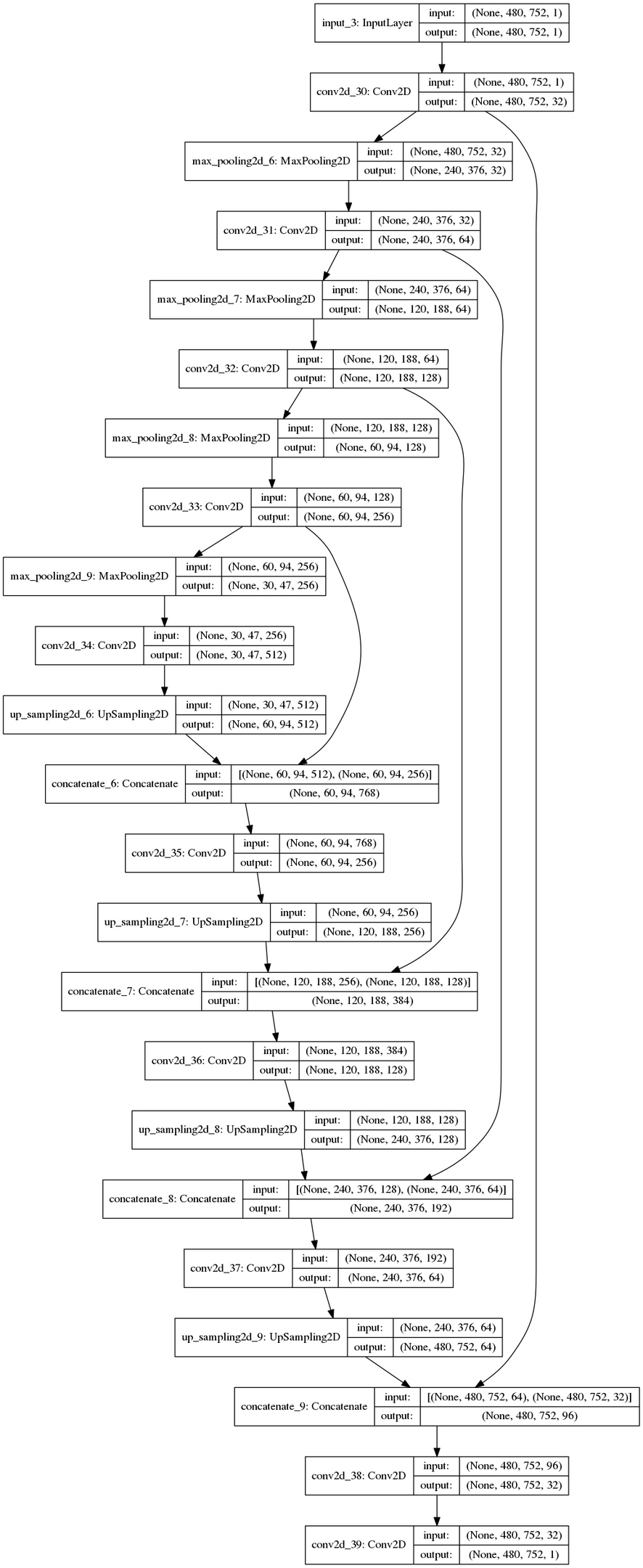


Рисунок 2. Архитектура первой нейронной сети для наложения маски.

|  |
| --- |
| model.compile(optimizer=Adam(lr=1e-4), loss=dice\_coef\_loss, metrics=[dice\_coef]) |

Листинг 1. С представлением параметров обучения

У оставшейся маски находятся левый верхний, левый нижний, правый нижний и правый верхний углы с помощью функции *car\_seg\_to\_plate\_and\_corners*.

Для обучения использовалась выборка с размеченными изображениями с 4мя точками, ктр представляют из себя угловые точки макси номера.

Затем, с плату геометрически преобразуется до размеров ширина = 200 пикселей, высота = 40 пикселей.



Рисунок 3. Пример преобразованной платы

Полученная плата передается на вход второй нейронной сети. 2-я нейросеть сегментирует символы прямоугольниками на изображении платы. (37

классов: bg, 0-9 a-z)



Рисунок 4. Пример размеченной платы

Цвет каждого прямоугольника зависит от класса символа. Background – 0, A-1…

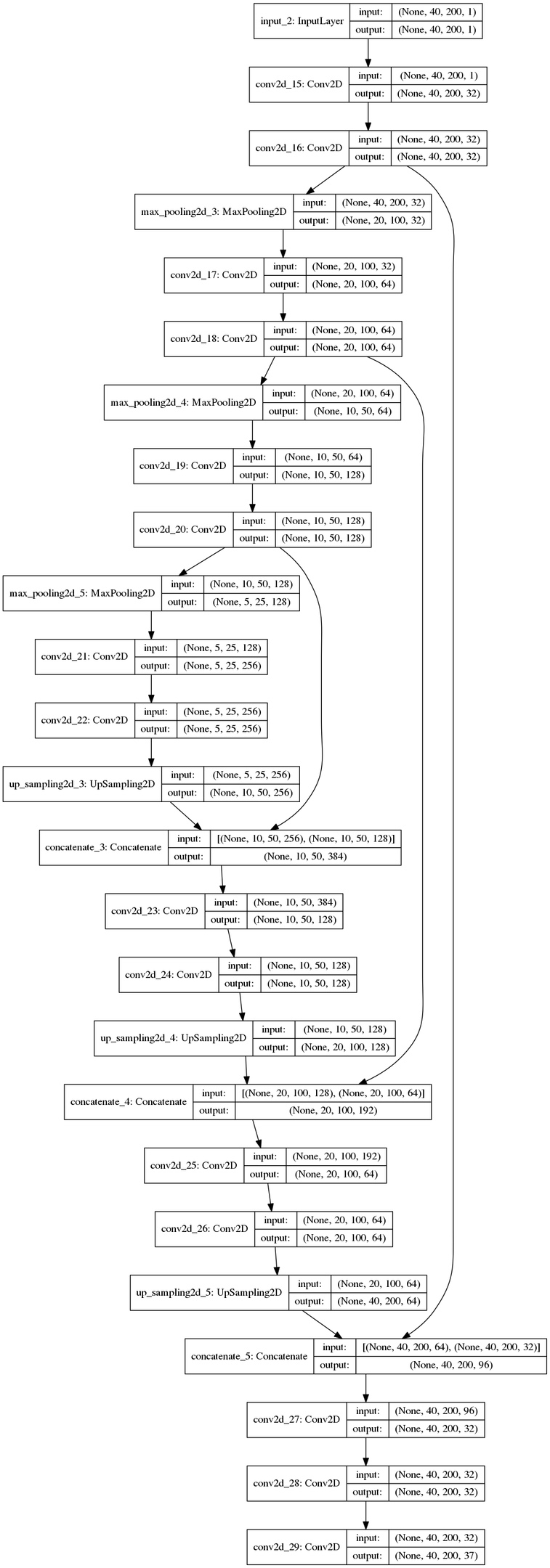


Рисунок 5. Архитектура второй сети

Полученное размеченное изображение преобразуется в текст с помощью функции plates\_seg\_to\_texts.

Последний шаг – преобразование выхода второй сети в строку. (numpy.ndarray shape=(?, 40,200, 37) ).

|  |
| --- |
| answer = {'status': 1, 'number':pred\_text, 'frame': self.corners\_to\_string(corners), 'confidence': 100} |

Листинг 2. Формат ответа

|  |
| --- |
| def predict(self, image):  #time.sleep(1)  #return '{"status": 1, "number": "AER964", "frame": "514,215;627,219;626,241;513,236", "confidence": 100}'  image = image / 255.0  image\_exp = self.expand(image)  mask\_exp = self.stage1.predict(image\_exp)  mask = self.squeeze(mask\_exp)  plate, corners = car\_seg\_to\_plate\_and\_corners(image, mask)  plate\_exp = self.expand(plate)  plate\_mask = self.stage2.predict(plate\_exp)  pred\_texts = plates\_seg\_to\_texts(plate\_mask)  pred\_text = pred\_texts[0] # Convert from ["text"] to "text" #TODO: change util  pred\_text = pred\_text.replace('#', '')  answer = {'status': 1, 'number':pred\_text, 'frame': self.corners\_to\_string(corners), 'confidence': 100}  answer = dumps(answer)  return answer |
| Листинг 3. Sekkar. Преобразование изображения в текст |

Необходимые файлы находятся /carplates\_server.

Для улучшения качества модели, необходимо сконцентрироваться на увеличении количества номеров формата, кроме как sss-nnn.

Также изменить архитектуру нейронной сети, ктр занимается распознаванием символов.

**Принцип работы с датасетом:**

принцип работы с датасетом:

- .csv файлы

- имя каждого файла:

[root\_dir]/[unique\_code]\_[postfix].png (пример:

/media/data/server\_img/analyze/15239652068218303\_plate.png)

- постфиксы:

“\_full” - full image

“\_plate” - transformed car plate

“\_mask” - full image mask

…

Все датасеты получаются путем генерации из изначального (полные изображения + images.csv)

**CARPLATES SERVER**

Структура проекта

api/ - function from lithuanian project that calls pg\_socket\_server

pentagon/ - server that serves image grabbing and saving on server. it makes requests to godfather -

computational server

godfather/ - computational server (with cuda compatible gpu)

truck\_weights/ - server that makes requests to godfather - computational server and saves images with

recognition in image name.

res/ - resources

utils/ - now there is only cipher class for safely transferring data over the net

z\_experiments - тесты

tests/ - unit tests

config.py - project scale settings

## Http interface:

### server on object asks:

* http POST request
* with attached file

json = {“shape”:(height,width), “cookie”:secret\_cookie} raw\_file = [json\_as\_bytes]\n[image\_as\_bytes] file\_to\_send = enrypt\_with\_aes(raw\_file)

### computational server answers:

if cookie is wrong:

* 401 Unauthorized
* {“error”:”Unknown user!”} else:

{

‘status’: 1, ‘number’: “AAA111”,

‘frame’: ‘310,285;441,289;441,317;311,313’, ‘confidence’: 100

}

## Структура базы данных

SHOTS\_TABLE\_NAME = ‘shots’

SHOTS\_FIELDS = [‘plateno’, ‘plate\_frame’, ‘plate\_image’, ‘full\_image’] example line:

CDCCC5 | 0,200;200,200;0,0;200,0 | | /tmp/car\_shots/2018/10/15/image\_00\_09\_07\_418687.png fields type - text

database engine - sqlite3

# SERVERS

# Установка на сервер:

* 1. systemd

cp myunit.service /etc/systemd/system/

systemctl daemon-reload # make system know that myunit.service was changed systemctl enable myunitsystemctl start myunit

systemctl status myunit

systemctl stop myunit- systemctl disable myunit

* + 1. python:
    2. anaconda
    3. virtual environment “tf\_env”
    4. source activate tf\_env
    5. absolute path: /home/interkodas/anaconda3/bin/python
    6. server in office: подключение:

app: ssh

cmd: ssh -X interkodas@88.119.96.115 -p 92

pass: Sinergija777

~/py\_safic - мои проекты, не трогать. нужно создать свою папку с проектами, например, py\_sergei

~/ - датасеты и папки с файлами, здесь хаос, но лучше его не увеличивать

server on object подключение: app: anydesk

login: interkodas-gb-bace-3000@ad

pass: Sinergija777

~/carplates\_server - наш проект

/var/www/weights/ - проект литовского программиста

/var/www/weights/var/media/shots/ - папка с фото

**HOWTOs**

### how to install opencv in anaconda

1. core func: conda install -c menpo opencv3
2. improvements: conda install -c conda-forge opencv

### how to install cuda and tensorflow in anaconda

1. check computer compatibility
2. install cuda toolkit 9.0
3. install cudnn (latest version) (x3 files) + check installation
4. install gpu drivers if not present
5. conda create tf\_env & pip install tensorflow-gpu (1.9)

### download dataset from server:

scp -P 92 -r interkodas@88.119.96.115:/home/interkodas/Desktop/carplates ./