# Разпознаване на ръкописно число чрез камера

Описание за това как работи алгоритъма.

Използваните библиотеки и кратко описание за тях.

Използваният сет от данни. (снимки)

Логика за тренирането на апликейшъна. Снимки и обяснение на кода. Точност и загуби.

Самият апликейшън. (снимки)

Курсовият проект трябва да има и документация, в която да се опише:

## каква е темата:

Разпознаване на ръкописно число чрез камера.

## какъв е алгоритъма за решаването на избрания проблем:

Adam optimization algorithm – това е алгоритъм за оптимизация, който е разширение на стохастично градиентно спускане. Наблюдава се широко прилагане на този алгоритъм в deep learning в компютърното зрение и обработката на естествен език.

Supervised learning – защото имаме достъп до дата сета.

## какви програмни технологии са използвани:

Python

OpenCV - <https://opencv.org/> <https://en.wikipedia.org/wiki/OpenCV>

## какви са възможностите, изискванията за изпозлване на приложението, ограниченията при използването му

## обобщение на експериментални резултати от гледна точка на точност при разпознаването.

Целта на курсовият проект е да се създаде програма, чрез която с помощта на уеб камерата на компютъра да се познае число показано на лист хартия. Използван език е python и алгоритъма за разпознаване е Adam.

**Training.py**

Този скрипт се използва за да обучим нашата програма с дадена колекция от стойности или в нашия случай - картинки с числа от 0 до 9. Първо четем и вмъкваме всички картинки на числа и техните наименования (лейбъли) и ги конвертираме към **numpy** масив, съдържащ лейбъл и картинки. Използваме numpy array, защото е по-бърз от обикновен list.

Text

Description automatically generated

След като вече имаме колекцията от лейбълнати числа използваме библиотеката sklearn и метода й train\_test\_split, чрез която ще разбъркаме числата и генерираме така наречените train колекции – X колекциите съдържат числата а Y техните лейбъли. Заделяме 80% от числата за обучителни цели и 20% за тестови цели. Правим същото разбъркване върху 80% от предишната колекция и за валидация – 80% обучение и 20% за валидация.

Text

Description automatically generated

С помощта на библиотеката OpenCV и нейния метод cvtColor() трансформираме карткините от колекциите в graysclae гамата, за по-добро разпознаване.

Text

Description automatically generated

Keras библиотеката пък ни предоставя клас ImageDataGenerator с помощта на който генерираме така наречени бачове от tensor image data, които представлява голяма колекция от елементи.

Text

Description automatically generated

От същата библиотека ползва и метода to\_categorical, който конвертира колекцията от векторните ни числа в матрица.

Text

Description automatically generated

Прилагаме и модела Sequential за да си генерираме наш модел на карткинките. Този модел e single layer, като има само по един input tensor и output tensor. Добавя няколко допълнителни layer-а към нашият модел, Conv2D, MaxPooling2D, Dropout, Flatten и Dense. Накрая оптимизираме чрез алгоритъма Adam от същата библиотека. Стартираме обучението чрез метода fit. Този метод обучава модел batch by batch.

Text

Description automatically generated

Използваме библиотеката matplotlib за да покажем графика с точността и загубата на нашият скрипт.

Text

Description automatically generated

И не на последно място запазваме създадения обучен модел, който ще използваме в програмата за разпознаване на число, чрез уеб камерата.



За да стартираме този обучителен процес е необходимо да имаме python 3-та версия. Навигираме през конзолата до скрипта и пускаме командата: python training.py

Получаваме следните графики за точност и загуба:

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

Материали:

* Numpy - <https://keras.io/api/utils/python_utils/#to_categorical-function>
* OpenCV - <https://docs.opencv.org/master/d1/dfb/intro.html>
* Sklearn - <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html>
* Sequential model - <https://keras.io/guides/sequential_model/> <https://faroit.com/keras-docs/1.2.0/getting-started/sequential-model-guide/>
* Training fit method - <https://keras.io/api/models/model_training_apis/#fit-method>
* Adam optimizer - <https://keras.io/api/optimizers/adam/> <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>
* Layers - <https://keras.io/api/layers/convolution_layers/convolution2d/> <https://keras.io/api/layers/pooling_layers/max_pooling2d/> <https://keras.io/api/layers/regularization_layers/dropout/> <https://keras.io/api/layers/reshaping_layers/flatten/> <https://keras.io/api/layers/core_layers/dense/>
* Matplotlib - <https://matplotlib.org/tutorials/introductory/pyplot.html>

Примерна тема 1:

Разпознаване на число през уеб камерата: https://www.youtube.com/watch?v=y1ZrOs9s2QA&ab\_channel=Murtaza%27sWorkshop-RoboticsandAI

Примерна тема 2:

Водене на регистър в общежитията, където се записват ЕГН-тата на посетителите и след това се експортват към база данни.

Примерна тема 3:

Разпознаване на номера на автомобили

В търговски център на влизане се записва номера на автомобила и така се следи колко време е престоя на този автомобил.

Nearest Neighbor Classifier (https://scikit-learn.org/stable/modules/neighbors.html)

A picture containing green, sign, street, city

Description automatically generated

A screenshot of a cell phone

Description automatically generated

Всяка снимка в сет-а, който ползваме е показана като масив от числа от 0 до 15.

A close up of a logo

Description automatically generated

За да намерим числото, което е подадено, трябва да намерим най-близкото разстояние, като сравним точките на новото число с точките на числата, които имаме от сета.A screenshot of a computer

Description automatically generatedA picture containing object, clock

Description automatically generated

Ако сравним числото с точка, която отговоря на 6, разстоянието ще бъде по-голямо.

A picture containing object, clock, drawing

Description automatically generated