ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Detekce objektů v ptačích hnízdech pomocí neuronových sítí
Student:	Jan Havlůj
Vedoucí:	Ing. Josef Pavlíček, Ph.D.
Studijní program:	Informatika
Studijní obor:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Navhněte a implementujte knihovnu umožňující zpracovávat data z kamery umístěné v hnízdě (ptačí budce) s cílem rozpoznat počet vajec v hnízdě. Pro rozpoznání použijte
existující algoritmy umělé intelegence využívající data získaná ze serveru PtaciOnline.cz a další sw. nástroje projektu BirdObserver (athena.pef.czu.cz).

Postupujte v těchto krocích:
1. Proveďte detailní specifickaci požadavků.
2. Seznamte se s projektem BirdObserver a strukturou dat na serveru PtáciOnline.cz.
3. Proveďte analýzu a návrh knihovny.
4. Návrh implementujte, zdokumentujte a vhodným způsobem otestujte.
5. Knihovnu navrhněte a implementujte v jazyce Java tak, aby bylo možné ji formou závislostí (dependency) integrovat s ostatními nástroji projektu BirdObserver.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 12. ledna 2017

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

Detekce objektů v ptačích hnízdech pomocí neuronových sítí

Jan Havlůj

Vedoucí práce: Ing. Josef Pavlíček, Ph.D.

30. prosince 2017

Poděkování

Chtěl bych poděkovat vedoucímu mé bakalářské práce, panu Ing. Josefovi Pavlíčkovi Ph.D. za jeho čas, ochotu a pomoc při vedení této práce. Děkuji svým rodičům a přítelkyni, kteří mi byli po celou dobu studia velkou oporou. V neposlední řadě bych chtěl poděkovat svým spolužákům, s kterými bylo celé studium příjemnější.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 30. prosince 2017 .....

Odkaz na tuto práci

Havlůj, Jan. *Detekce objektů v ptačích hnízdech pomocí neuronových sítí.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

V několika větách shrňte obsah a přínos této práce v češtině. Po přečtení abstraktu by se čtenář měl mít čtenář dost informací pro rozhodnutí, zda chce Vaši práci číst.

Klíčová slova neuronové sítě, počítačové vidění, detekce objektů, rozpoznávání obrazu, detekce vajec, Java knihovna

Abstract

Sem doplňte ekvivalent abstraktu Vaší práce v angličtině.

Keywords neural networks, computer vision, object detection, image recognition, egg detection, Java library

Obsah

Seznam obrázků

Úvod

asdfasfsdf

Kapitola 1

Cíl práce

1.1 Nefunkční požadavky

1.2 Funkční požadavky

Text

Kapitola 2

Rešerše

2.1 Počítačové vidění

2.2 Segmentace obrazu

2.3 Detekce objektů

2.4 Klasifikace obrazu

2.5 Neuronové sítě

2.5.1 CNN

2.5.2 RCNN

2.6 Tensorflow

Text

Kapitola 3

Analýza a návrh

3.1 Technologie

3.2 Funkční požadavky

3.3 Rozpoznávání obrazu

3.4 Detekce objektů

3.5 Shrnutí kapitoly

Text

Kapitola 4

Implementace rozpoznávání obrazu

4.1 Volba technologií

4.2 Trénovací data

4.3 Nástroje

4.4 Příprava dat

4.5 Trénování neuronové sítě

4.6 Ověření funkčnosti

Text

Kapitola 5

Volba řešení

5.1 Srovnání implementací

5.2 Exportování grafu

5.3 Implemetace Java knihovny

Text

Kapitola 6

Ověření implementace

6.0.1 Metodika vývoje

6.0.2 Testování

6.0.3 Možnosti vylepšení

Text

Závěr

Literatura

Příloha **A**

Seznam použitých zkratek

GUI Graphical user interface

XML Extensible markup language

HTML HyperText Markup Language

RAM Random-access memory

PDF Portable Document Format

NN Neural Network

CNN Convolutional Neural Network

RCNN Recursive Convolutional Neural Network

API Application Programming Interface

Příloha B

Dokumentace API knihovny

Příloha obsahuje dokumentaci API Java knihovny, která je výsledkem této práce. Dokumentace byla vygenerována ze zdrojových kódů nástroji javadoc[1] a TexDoclet[2]. Kód je psán v anglickém jazyce, stejně jako jeho dokumentace. Proto je text i zde v angličtině.

B.1 Package org.cvut.havluja1.eggdetector

Package Contents *Page*

Classes

B.1.1 Class EggDetector

B.1.1.1 Count the number of eggs in given images

The egg detector is a library that helps you count the number of eggs in a given folder. Egg detector works by using TeonsorFlow Object Detection API in the background. To learn more, see https://www.tensorflow.com.

Example usage:

```
EggDetector eggDetector = new EggDetector();
SequenceClassifier sequenceClassifier = eggDetector.evaluate(new File("image_dir"));
System.out.println("final count: " + sequenceClassifier.getFinalCount());
System.out.println("individual scores: " + sequenceClassifier.getIndividualCounts());
eggDetector.closeSession();
```

[1] http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html
[2] http://doclet.github.io/

B.1.1.2 Declaration

```
public class EggDetector
 extends java.lang.Object
```

B.1.1.3 Constructor summary

EggDetector()
Constructor loads the pre-trained frozen graph into memory.

B.1.1.4 Method summary

closeSession()
Closes the EggDetector session.

evaluate(File)
Runs egg detection on a given *dir*.

getMinimalConfidence()
Get the minimalConfidence setting for this instance.

isDebugMode()
Get this instance's debug mode setting.

setDebugMode(boolean)
Set this instance's debug mode setting.

setMinimalConfidence(float)
Set the minimalConfidence setting for this instance.

toString()

B.1.1.5 Constructors

- *EggDetector*

  ```
  public EggDetector()
  ```

  - **D**escription
    Constructor loads the pre-trained frozen graph into memory.
    It also checks whether TensorFlow is supported on your platform.

B.1.1.6 Methods

- *closeSession*

  ```
  public void closeSession() throws java.lang.IllegalStateException
  ```

  - **D**escription

    Closes the EggDetector session. This instance of EggDetector will not be usable again.

  - **T**hrows
    * `java.lang.IllegalStateException` – if the session has been closed already by calling `closeSession()`

- *evaluate*

```
public SequenceClassifier evaluate(java.io.File dir) throws
    java.lang.IllegalArgumentException, java.lang.IllegalStateException
```

  - **D**escription

    Runs egg detection on a given *dir*.

  - **P**arameters
    * `dir` – a directory containing .jpg or .png files for object detection

  - **R**eturns –

  - **T**hrows
    * `java.lang.IllegalArgumentException` – if *dir* is not a directory or contains no images
    * `java.lang.IllegalStateException` – if the session has been closed already by calling `closeSession()`

- *getMinimalConfidence*

```
public float getMinimalConfidence()
```

  - **D**escription

    Get the *minimalConfidence* setting for this instance.

    Minimal Confidence score is used as a confidence boundary during the process of object detection. An object that has been detected with a confidence score lower than *minimalConfidence* is ignored. An object that has been detected with a confidence score higher or equal than *minimalConfidence* is added to the final result list.

  - **R**eturns – This instance's *minimalConfidence* setting.

- *isDebugMode*

```
public boolean isDebugMode()
```

  - **D**escription

    Get this instance's debug mode setting.

    If debug mode is enabled (set to true), the library will open a `JFrame` for each processed image with detections graphically highlighted.

– **R**eturns – debug mode setting for this instance

- *setDebugMode*

```
public void setDebugMode(boolean debugMode) throws
    java.lang.IllegalStateException
```

– **D**escription

Set this instance's debug mode setting.

If debug mode is enabled (set to true), the library will open a `JFrame` for each processed image with detections graphically highlighted.

– **P**arameters

∗ `debugMode` – turn the debug mode on or off

– **T**hrows

∗ `java.lang.IllegalStateException` – if the session has been closed already by calling `closeSession()`

- *setMinimalConfidence*

```
public void setMinimalConfidence(float minimalConfidence) throws
    java.lang.IllegalStateException
```

– **D**escription

Set the *minimalConfidence* setting for this instance.

Minimal Confidence score is used as a confidence boundary during the process of object detection. An object that has been detected with a confidence score lower than *minimalConfidence* is ignored. An object that has been detected with a confidence score higher or equal than *minimalConfidence* is added to the final result list.

– **P**arameters

∗ `minimalConfidence` – *minimalConfidence* for this instance

– **T**hrows

∗ `java.lang.IllegalStateException` – if the session has been closed already by calling `closeSession()`

- *toString*

```
public java.lang.String toString()
```

B.1.2 Class SequenceClassifier

B.1.2.1 A class containing object detection results for a given directory

SequenceClassifier is a data class containing the results of object detection for a given directory. When constructed, object detection is performed on all images and results are stored in memory.

Example usage:

```
EggDetector eggDetector = new EggDetector();
SequenceClassifier sequenceClassifier = eggDetector.evaluate(new File("image_dir"));
System.out.println("final count: " + sequenceClassifier.getFinalCount());
System.out.println("individual scores: " + sequenceClassifier.getIndividualCounts());
eggDetector.closeSession();
```

B.1.2.2 Declaration

```
public class SequenceClassifier
 extends java.lang.Object
```

B.1.2.3 Method summary

getFinalCount()
Get the final score for the entire directory.

getIndividualCounts()
Gets the individual egg count for every image provided.

B.1.2.4 Methods

- *getFinalCount*

  ```
  public java.lang.Integer getFinalCount()
  ```

  - **D**escription

    Get the final score for the entire directory.

    The final score is calculated as follows:
    * individual scores of images are sorted and counted
    * the highest egg count is returned as a result if we detected this egg count in at least two different images
    * if no two images contain the same egg count, the highest detected egg count is returned
    * if no eggs are detected in any of the images, 0 is returned
  - **R**eturns – final egg count for this instance

- *getIndividualCounts*

  ```
  public java.util.Map getIndividualCounts()
  ```

- **D**escription
  Gets the individual egg count for every image provided.
- **R**eturns – A map of individual scores. The key is the filename. The value is the egg count.

Příloha C

Použité programy

TexStudio psaní bakalářské práce v LaTeXa překlad do PDF

Notepad++ úprava textových souborů

gedit úprava textových souborů

IntelliJ IDEA Ultimate psaní implementace v jazyce Java

GIT verzování kódu

JDK8 kompilace, ladění a testování Java knihovny

Python 3 zpracování dat, trénování neuronových sítí

Tensorflow softwarová knihovna pro strojové učení

LabelImg označování trénovacích dat

javadoc generování dokumentace do HTML

TexDoclet generování dokumentace do LaTeX

wget hromadné stahování dat

bash pomocné skripty pro zautomatizování práce

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `bin` ..... adresář se spustitelnou formou implementace
- `doc` ..... adresář s dokumentací implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `eggdetector` ..... zdrojové kódy implementace knihovny
    - `nn_training` ..... zdrojové kódy pro trénování neuronové sítě
    - `tools` ..... zdrojové kódy pomocných nástrojů k tvorbě bakalářské práce
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF