SPECIFIKACE ROČNÍKOVÉHO PROJEKTU

Klasifikace muzejních umění pomocí konvoluční neuronové sítě

Jáchym Mraček

Obsah

Jméno	Datum	Důvod změny	Verze

Základní informace

Popis softwarového díla

Softwarové dílo se zaměřuje na klasifikaci muzejních objektů z několika fotek. Klasifikace proběhne pomocí metod strojového učení v počítačovém vidění a zpracování digitálního obrazu. Naše dílo bude zpracovávat objekty pomocí konvoluční neuronové sítě, která bude přizpůsobená naším objektům.

Použité technologie

Program bude napsán pomocí několika knihoven, které vytvářejí vrstvy konvoluční neuronové a sítě a importují trénovací model **Tensorflow.** Softwarové dílo také využívá knihovnu **OpenCV** pro zpracování vstupních dat pomocí příkazu **cv2.imread()** a knihovnu **Numpy** používáme k manipulaci s obrázky a knihovnu **Matplotlib** je knihovna pro zobrazování obrázků.

Vývojové prostředí

Program je napsán ve vývojovém prostředí **Visual Studio Code**.

Odkazy

TensorFlow - https://www.tensorflow.org/ OpenCV - https://opencv.org/

Stručný popis řešení softwarového díla

Data

Rozdělení dat

Data jsou rozdělena do tří skupin:

- Trénovací
- Validační
- Testová (předem určené)

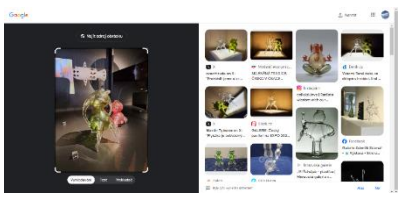
Naše úloha bude obsahovat 100 tříd různých muzejních objektů, které náš program musí rozeznat.

Jak zvolit trénovací data a validační data

Pro každou ze 100 tříd stáhneme z Googlu všechny podobné obrázky pomocí vyhledání shodnosti testového obrázku (reprezentující třídu muzejního objektu)

Stažené obrázky budou rozděleny do validačních a testových dat, tak

že 70 procent budou trénovací data a 30 procent budou validační data.



Úprava stažených obrázků

Stažené obrázky upravíme pomocí segmentace hlavního obrázku, tak aby pozadí obrázku neovlivňovalo klasifikaci a trénování konvoluční neuronové sítě.

Uložení dat

Data budou uloženy do tří složek, které jsou:

- **trainingData** (složka, která obsahuje dalších 100 složek s trénujícími obrázky popisující příslušnou třídu muzejnícho objektu)
- **validateData** (složka, která obsahuje dalších 100 složek s validačními obrázky popisující příslušnou třídu muzejnícho objektu)
- **testingData** (Zadaná vstupní sada obrázků)

Načtení dat do programu

Pro správné načtení dat použijeme příkaz: **tf.keras.utils.image_dataset_from_directory(„traininfClasses“)**

Pro načtení trénovacích obrázku použijeme příkaz: `data_withTag=data.as_numpy_iterator()`
Kde každému obrázku přiradíme štítek, který reprezentuje.

Normalizace dat

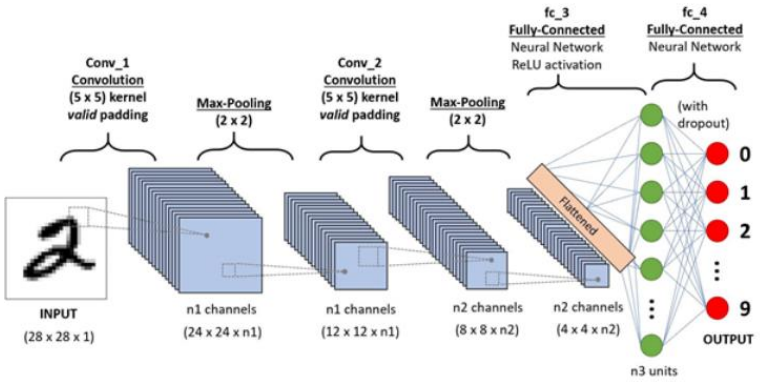
Data-obrázky znormalizujeme do (0,1), tedy velikosti jednotlivých RGB vydělíme 255.
normalized_image=(cv2.imread(„path“).astype(np.float32) / 255)

Konvoluční neuronová síť

Konvoluční neuronovou síť importujeme pomocí knihovny TensorFlow a architekturu konvoluční neuronové sítě sestavíme pomocí **tensorflow.keras.layers** ze které importujeme všechny vrstvy, které tvoří konvoluční neuronovou síť.

Konvoluční neuronová síť je tvořena:

- konvolučními vrstvy — **model.add(Conv2D(, (),, activation='relu', input_shape=()))**
- MaxPooling vrstvy — **model.add(MaxPooling2D())**
- aktivační funkcí (Relu)
- plně propojenou vrstvou — **model.add(Dense(, activation='relu'))**



Trénování

Pro začátek trénování musí model zkompilovat pomocí příkazu:

```
model.compile('adam',loss=tf.losses.BinaryCrossentropy(),metrics=['accuracy'])
```

Potom konvoluční síť natrénujeme pomocí příkazu:

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir="log")
model.fit(train, epochs=,validation_data=val,callbacks=[tensorboard_callback])
```

Filtry

Naše konvoluční síť obsahuje konvoluční vrstvy, které jsou založené na konvolučních filtrech. Filtry budou zvoleny, tak aby co nejlépe zachytili klíčové prvky příslušných tříd, které se zaměřují na rohy, hrany, tvary a podobně.

Vyhodnocení

Závěrem vyhodnotíme nejlepší volbu konvoluční sítě s konvolučními filtry. Program bude hodnocen pomocí příkazů:

- **predictian= model.predict(X)**
- **pre.update_state(y,predictian)**
- **print(pre.result())**

Funkce programu a vnější rozhraní

Funkce

Uživatel bude moci:

- nahrát složku s obrázky, které chce klasifikovat

Program po spuštění určí, do jaké třídy vstupní obrázky patří, tak že na okně napíše jméno obrázku a příslušnou třídu do, které byl obrázek klasifikovaný.

Operační systém

Program je naprogramován pro operační systém Windows.

Spouštění programu

Program se spouští přes příkazovou řádku, kde příkaz nemá žádné parametry

> *Python klasifikace:muzejnich_umeni.py*

Po spuštění uživatel uvidí dialogové okno.

Dialogové okno

Dialogové okno je naprogramována pomocí knihovny **Tkinter,** kde úvodní okno obsahuje tlačítko start a konec. Pokud uživatel stiskne start, pak se uživatel dostane do nového

okna, kde je možnost zvolit složku s obrázky, které uživatel chce klasifikovat. Po zadání cesty uživatel stiskne tlačítko klasifikovat a následně se zobrazí na vyhodnocovací ploše příslušné klasifikace.

Pokud uživatel chce opět klasifikovat, pak pouze změní cestu a opět stiskne tlačítko klasifikovat.

Pokud uživatel bude chtít aplikaci ukončit stiskne zpět, kde se dostane do úvodního okna anebo konec, čímž se okno okno zavře a program skončí.

Detailní popis funkcionality

Funkcionalita a operační systém

V případě správného zadání příkazu se na obrazovce zobrazí dialogové okno, kde uživatel může zadávat vstupní cestu s obrázky. Program je dělán pro operační systém Windows.

Chybové hlášky

Pokud uživatel zadá chybnou cestu k složce s obrázky, pak se na vyhodnocovací ploše zobrazí hláška „zadaná cesta není správná“ následně pro správnou klasifikaci uživatel musí zadat správnou cestu.

Pokud složka nepůjde otevřít nebo vznikne jiný problém, pak se na vyhodnocovacím okně zobrazí zpráva „nastala chyba při zpracování dat“.

Motivační příklad užití a výstupy

Uživatel otevře terminál a pomocí příkazu > *Python klasifikace:muzejnich_umeni.py* spustí úvodní okno, kde potom stiskne start, čímž se zobrazí nové okno ve kterým je pole pro zadání cesty ke složce s obrázky, kterou chce uživatel klasifikovat. Po zadání stiskne klasifikovat a na vyhodnocovacím=výstupním okně se zobrazí např: obr1.png židle
obr2.png stůl

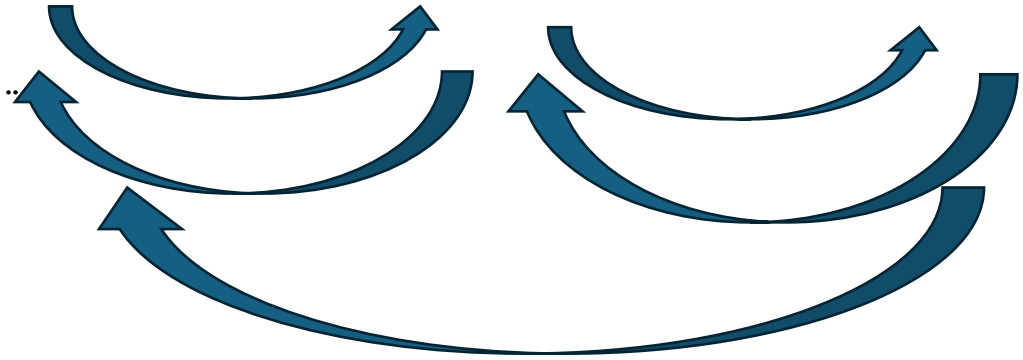
atd

Následně uživatel stisknutím tlačítka konec opustí okno a program se ukončí.

Obrazovky

Program bude obsahovat dvě základní obrazovky – úvodní a klasifikační, kde úvodní má dvě tlačítka start a konec a klasifikační má pole pro zadání cesty s obrázky a tři tlačítka klasifikovat a konec

TERMINÁL ÚVODNÍ OBRAZOVKA KLASIFIKAČNÍ OBRAZOVKA



Time-line

Datum	*Milník*	*Způsob prezentace*
1.6.2024	finální verze této specifikace	Existující dokumentace
31.7.2024	Předvedení programu	Osobní prezentace+kod v gitlabu
15.8.2024	finální verze programu	Osobní prezentace+kod v gitlabu

Zdroje

https://d3s.mff.cuni.cz/files/teaching/nprg045/SablonaSpecifikacePriklad.pdf

https://github.com/nicknochnack/ImageClassification/blob/main/Getting%20Started.ipynb

https://studuj.digital/machine-learning-slovnik-pojmu/